Cisco SD-WAN (Viptela) Configuration Guide, Release 18.1

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

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

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

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

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

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

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

Release Notes

Release Notes for Release 18.1

These release notes accompany Viptela Software Release 18.1, for Release 18.1.1. The Viptela software runs on all Viptela devices, including vSmart controllers, vEdge routers, vBond orchestrators, and vManage NMSs. Release 18.1.1 is available for limited use only.

Viptela Software Release 18.1
March 30, 2018
Revision 1

Product Features

Below are the main product features in Viptela Software Release 18.1:

- **Bootloader** —On all MIPS-based vEdge routers, the bootloader (u-boot) security has been enhanced.
- **Cisco ISR 4331 router** —The Viptela software runs on the Cisco ISR 4331 router.
- **HTTPS service on vEdge router tunnel interfaces** —By default, HTTPS service is allowed on vEdge router tunnel interfaces. *See allow-service and Software Caveats, below.*
- **IPv6 enhancements** —vEdge routers support IPv6 Router Advertisement (RA) and Router Solicitation (RS) messages, and IPv6 stateless autoconfiguration (SLAAC). Using RA messages, DHCPv6 learns the default route. RA messages also allow the router to advertise DNS information to IPv6 hosts. RA and RS messages and SLAAC allow zero-touch provisioning (ZTP) over IPv6.
- **Root access** —On vBond orchestrators, vEdge routers, and vSmart controllers, root access has been disabled.

Command Changes

New and Modified Configuration Commands

Command	Hierarchy	New	Modified	Comments
allow-service	vpn 0 tunnel interface		X	Allow HTTPS for vEdge routers.

New and Modified Operational Commands

None

REST API Changes

- Under Configuration - Device Template, the template/device/config/validate call has been renamed to template/device/config/verify.

Upgrade to Release 18.1

For details on upgrading the Viptela software, see Software Installation and Upgrade .

If your vManage NMS is running as a cluster, please contact Customer Support for assistance when upgrading to Release 18.1.0 or later.

To upgrade to Release 18.1:

1. In vManage NMS, select the Maintenance ► Software Upgrade screen.
2. Upgrade the controller devices to Release 18.1 in the following order:
    a. First, upgrade the vManage NMSs in the overlay network.
    b. Then, upgrade the vBond orchestrators.
    c. Next, upgrade the vSmart controllers.
3. Select the Monitor ► Network screen.
4. Select the devices you just upgraded, click the Control Connections tab, and verify that control connections have been established.
5. Select the Maintenance ► Software Upgrade screen, and upgrade the vEdge routers.

Note: After you upgrade software on a vManage NMS to any major release, you can never downgrade it to a previous major release. For example, if you upgrade the vManage NMS to Release 18.1, you can never downgrade it to Release 17.2 or to any earlier software release.

The major release number consists of the first two numbers in the software release number. For the Viptela software, 17.2 and 18.1 are examples of major releases. Releases 17.2.0 and 18.1.0 denote the initial releases, and Releases 17.2.7 and 18.1.1 are maintenance releases.

Upgrade from Release 16.2 and Earlier Software Releases

Because of software changes in Release 16.3, you must modify the router configuration as follows before you upgrade from Release 16.2 or earlier to Release 18.1:

- You can no longer configure RED drops on low-latency queuing (LLQ; queue 0). That is, if you include the **policy qos-scheduler scheduling llq** command in the configuration, you cannot configure **drops red-drop** in the same QoS scheduler. If your vEdge router has this configuration, remove it before upgrading to Release 18.1. If you do not remove the RED drop configuration, the configuration process (confd) will fail after you perform the software upgrade, and the Viptela devices will roll back to their previous configuration.
- For vEdge 2000 routers, you can no longer configure interfaces that are not present in the router. That is, the interface names in the configuration must match the type of PIM installed in the router. For example, if the PIM module in slot 1 is a 10-Gigabit Ethernet PIM, the configuration must refer to the proper interface name, for example, **10ge1/0** , and not **ge1/0** . If the interface name does not match the PIM type, the software upgrade will fail. Before you upgrade from Release 16.2 or earlier to Release 18.1, ensure that the interface names in the router configurations are correct.

Caveats

Hardware Caveats

The following are known behaviors of the Viptela hardware:

- On vEdge 1000 routers, support for USB controllers is disabled by default. To attach an LTE USB dongle to a vEdge 1000 router, first attach the dongle, and then enable support for USB controllers on the vEdge router, by adding the **system usb-controller** command to the configuration. When you enter this command in the configuration, the router immediately reboots. Then, when the router comes back up, continue with the router configuration. Also for vEdge 1000 routers, if you plug in an LTE USB dongle after you have enabled the USB controller, or if you hot swap an LTE USB dongle after you have enabled the USB controller, you must reboot the router in order for the USB dongle to be recognized. For information about enabling the USB controller, see USB Dongle for Cellular Connection .

- For vEdge 2000 routers, if you change the PIM type from a 1-Gigabit Ethernet to a 10-Gigabit Ethernet PIM, or vice versa, possibly as part of an RMA process, follow these steps:
  1. Delete the configuration for the old PIM (the PIM you are returning as part of the RMA process).
  2. Remove the old PIM, and return it as part of the RMA process.
  3. Insert the new PIM (the PIM you received as part of the RMA process).
  4. Reboot the vEdge 2000 router.
  5. Configure the interfaces for the new PIM.
- On a vEdge 5000 router, you cannot enable TCP optimization by configuring the **tcp-optimization-enabled** command.

Software Caveats

The following are known behaviors of the Viptela software:

Cellular Interfaces

- The vEdge 100wm router United States certification allows operation only on non-DFS channels.
- When you are configuring primary and last-resort cellular interfaces with high control hello interval and tolerance values, note the following caveats:
  - When you configure two interfaces, one as the primary interface and the other as the last-resort interface, and when you configure a high control hello interval or tolerance values on the last-resort interface (using the **hello-interval** and **hello-tolerance** commands, respectively, the OMP state indicates init-in-gr even though it shows that the control connections and BFD are both Up. This issue was resolved in Release 16.2.3. However, the following caveats exist:
    — You can configure only one interface with a high hello interval and tolerance value. This interface can be either the primary or the last-resort interface.
    — In certain cases, such as when you reboot the router or when you issue shutdown and no shutdown commands on the interfaces, the control connections might take longer than expected to establish. In this case, it is recommended that you issue the **request port-hop** command for the desired color. You can also choose to wait for the vEdge router to initiate an implicit port-hop operation. The **request port-hop** command or the implicit port hop initiates the control connection on a new port. When the new connection is established, the stale entry is flushed from the vSmart controllers.
  - If the primary interface is Up, as indicated by the presence of a control connection and a BFD session, and if you configure a last-resort interface with higher values of hello interval and tolerance than the primary interface, if you issue a **shutdown** command, followed by a **no shutdown** command on the last-resort interface, the last-resort interface comes up and continuously tries to establish control connections. Several minutes can elapse before the operational status of the last-resort interfaces changes to Down. If this situation occurs, it is recommended that you issue a **request port-hop** command for the desired color.
  - If you have configured a primary interface and a last-resort interface that has higher hello interval and tolerance values than the primary interface, and if the last-resort interface has control connections to two vSmart controllers, if you issue a **shutdown** command, followed by a **no shutdown** command on the last-resort interface, a control connection comes up within a reasonable amount of time with only one of the vSmart controllers. The control connection with the second vSmart controller might not come up until the timer value configured in the hello tolerance has passed. If this situation occurs, it is recommended that you issue a **request port-hop** command for the desired color.
- When you activate the configuration on a router with cellular interfaces, the primary interfaces (that is, those interfaces not configured as circuits of last resort) and the circuit of last resort come up. In this process, all the interfaces begin the process of establishing control and BFD connections. When one or more of the primary interfaces establishes a TLOC connection, the circuit of last resort shuts itself down because it is not needed. During this shutdown process, the circuit of last resort triggers a BFD TLOC Down alarm and a Control TLOC Down alarm on the vEdge router. These two alarms are cleared only when all the primary interfaces lose their BFD connections to remote nodes and the circuit of last resort activates itself. This generation and clearing of alarms is expected behavior.
- For cellular interface profile, the profile number can be 0 through 15. Profile number 16 is reserved, and you cannot modify it.

Configuration and Command-Line Interface

- When you issue the **request reset configuration** command on a vEdge Cloud router, a vManage NMS, or a vSmart controller, the software pointer to the device's certificate might be cleared even though the certificate itself is not deleted. When the device reboots and comes back up, installation of a new certificate fails, because the certificate is already present. To recover from this situation, issue the **request software reset** command.

Control and BFD Connections

- When a vBond orchestrator, vManage NMS, or vSmart controller goes down for any reason and the vEdge routers remain up, when the controller device comes back up, the connection between it and the vEdge router might shut down and restart, and in some cases the BFD sessions on the vEdge router might shut down and restart. This behavior occurs because of port hopping: When one device loses its control connection to another device, it port hops to another port in an attempt to reestablish the connection. For more information, see the Firewall Ports for Viptela Deployments article. Two examples illustrate when this might occur:
  - When a vBond orchestrator goes down for any reason, the vManage NMS might take down all connections to the vEdge routers. The sequence of events that occurs is as follows: When the vBond orchestrator crashes, the vManage NMS might lose or close all its control connections. The vManage NMS then port hops, to try to establish connections to the vSmart controllers on a different port. This port hopping on the vManage NMS shuts down and then restarts all its control connections, including those to the vEdge routers.
  - All control sessions on all vSmart controllers go down, and BFD sessions on the vEdge routers remain up. When any one of the vSmart controllers comes back up, the BFD sessions on the routers go down and then come back up because the vEdge routers have already port hopped to a different port in an attempt to reconnect to the vSmart controllers.
- When a vEdge router running Release 16.2 or later is behind a symmetric NAT device, it can establish BFD sessions with remote vEdge routers only if the remote routers are running Release 16.2 or later. These routers cannot establish BFD sessions with a remote vEdge router that is running a software release earlier than Release 16.2.0.
- When you add or remove an IPv4 address on a tunnel interface (TLOC) that already has an IPv6 address, or when you add or remove an IPv6 address on a TLOC that already has an IPv4 address, the control and data plane connections for that interface go down and then come back up.
- Release 16.3 introduces a feature that allows you to configure the preferred tunnel interface to use to exchange traffic with the vManage NMS. In the vManage NMS, you configure this on cellular, Ethernet, and PPP Interface feature templates, in the vManage Connection Preference field under Tunnel Interface. In the CLI, you configure this with the **vmanage-connection-preference** command. The preference value can be from 0 through 8, with a lower number being more preferred. The default value is 5. If you set the preference value to 0, that tunnel interface is never used to exchange traffic with the vManage NMS, and it is never able to send or receive any overlay network control traffic.
  With this configuration option, there is one situation in which you can accidentally configure a device such that it loses all its control connections to all Viptela controller devices (the vManage NMSs and the vSmart controllers). If you create feature templates and then consolidate them into a device template for the first time, the NMS software checks whether each device has at least one tunnel interface. If not, a software error is displayed. However, when a device template is already attached to a device, if you modify one of its feature templates such that the connection preference on all tunnel interfaces is 0, when you update the device with the changes, no software check is performed, because only the configuration changes are pushed to the device, not the entire device template. As a result, these devices lose all their control connections. To avoid this issue, ensure that the vManage connection preference on at least one tunnel interface is set either to the default or to a non-0 preference value.

Installation

- One of the steps in installing signed certificates on vEdge Cloud routers is to generate a bootstrap configuration file. On a vManage NMS running Release 18.1.1 or later, if you generate the bootstrap configuration file for a vEdge Cloud router that is running software earlier than Release 18.1.1, the generated configuration will fail on that router. This is because in Release 18.1.1, a new default configuration, **allow-service https** , has been added for vEdge tunnel interfaces. Earlier software releases do not support the HTTPS service on vEdge router tunnel interfaces. As a workaround, edit the generated bootstrap configuration file to remove the HTTPS configuration before you attach the configuration to the router.

Interfaces

- On virtual interfaces, such as IRB, loopback, and system interfaces, the duplex and speed attributes do not apply, and you cannot configure these properties on the interfaces.

- Traffic flow on IPsec tunnels might be interrupted when you configure only tunnel interface parameters, such as MTU and dead-peer detection. [VIP-31426]
- When a vEdge router has two or more NAT interfaces, and hence two or more DIA connections to the internet, by default, data traffic is forwarding on the NAT interfaces using ECMP. To direct data traffic to a specific DIA interface, configure a centralized data policy on the vSmart controller that sets two actions— **nat** and **local-tloc** color. In the **local-tloc** color action, specify the color of the TLOC that connects to the desired DIA connection.

IPv6

- You can configure IPv6 only on physical interfaces (**ge** and **eth** interfaces), loopback interfaces (**loopback0** , **loopback1** , and so on), and on subinterfaces (such as **ge0/1.1**).
- For IPv6 WAN interfaces in VPN 0, you cannot configure more than two TLOCs on the vEdge router. If you configure more than two, control connections between the router and the Viptela controllers might not come up.
- IPv6 transport is supported over IPsec encapsulation. GRE encapsulation is not supported.
- You cannot configure NAT and TLOC extensions on IPv6 interfaces.
- DHCPv6 returns only an IPv6 address. No default information is accepted. IPv6 router solicitation and router advertisement messages are not processed.

IRB

- On integrated routing and bridging (IRB) interfaces, you cannot configure autonegotiation .

NAT

- When you reboot a vSmart controller, the BFD sessions for all symmetric NAT devices go down and come back up. This is expected behavior.

Security

- It is recommended that you use IKE Version 2 only with Palo Alto Networks and Ubuntu strongSwan systems only. Viptela has not tested IKE Version 2 with other systems.

SNMP

- When you configure an SNMP trap target address, you must use an IPv4 address.
- The Viptela interface MIB supports both 32-bit and 64-bit counters, and by default sends 64-bit counters. If you are using an SNMP monitoring tool that does not recognize 64-bit counters, configure it to read 32-bit MIB counters.
- On a vEdge router, if you perform an snmpwalk getnext request for an OID for which there is no information, the response that is returned is the next available instance of that OID. This is the expected behavior.

System

- When a task stops and a vEdge router reboots, the router might no longer reboot. This situation occurs after the router reboots three times within 20 minutes, five times within 60 minutes, or seven times within the last 24 hours. During this time, the control plane on the router remains up, so traffic continues to be sent to the node. To override this behavior, recover the router via the console port.
- Pushing a device configuration template to a vEdge router might fail because of a bridge configuration validation failure. This issue occurs when a bridge with VLAN and interfaces is already configured on the router and the template being pushed modifies these parameters. As a workaround, copy the template, delete the entire bridge configuration, and push the template to the router. Then add the original bridge configuration to the template, and push that template to the router.

Virtual Machines

- For a vEdge Cloud VM instance on the KVM hypervisor, for Viptela Releases 16.2.2 and later, it is recommended that you use virtio interfaces. For software versions earlier than Release 16.2.2, if you are using the Ubuntu 14.04 or 16.04 LTS operating system, you can use IDE, virtio, or virtio-scsi interfaces.

vManage NMS

- On a Viptela device that is being managed by a vManage NMS system, if you edit the device's configuration from the CLI, when you issue the **commit** command, you are prompted to confirm the commit operation. For example:
  vEdge(config-banner)# **commit**
  The following warnings were generated:
  'system is-vmanaged': This device is being managed by the vManage. Any configuration changes to this device will be overwritten by the vManage.
  Proceed? [yes,no]
  You must enter either **yes** or **no** in response to this prompt.
  During the period of time between when you type commit and when you type either **yes** or **no** , the device's configuration database is locked. When the configuration database on a device is locked, the vManage NMS is not able to push a configuration to the device, and from the vManage NMS, you are not able to switch the device to CLI mode.

- The members of a vManage cluster rely on timestamps to synchronize data and to track device uptime. For this time-dependent data to remain accurate, do not change the clock time on any one of the vManage servers of the cluster after you create the cluster.

- When you use the vManage Maintenance ► Software Upgrade screen to set the default software version for a network device, that device must be running Release 16.1 or later at the time you set the default software version. If the network device is running Release 15.4 or earlier, use the CLI **request software set-default** command to set the default software version for that device.

- When you are using a vManage cluster, when you are bringing up new vManage NMS in the cluster, use an existing vManage NMS to install the certificate on the new vManage NMS.

- In vManage feature configuration templates, for the passwords listed below, you cannot enter a cleartext password that starts with $4 or $8. You can, however, use such passwords when you are configuring from the CLI.
  - Neighbor password, in the BGP feature configuration template
  - User password, in the Cellular Profile feature configuration template
  - Authentication type password and privacy type password, in the SNMP feature configuration template
  - RADIUS secret key and TACACS+ secret key, in the System feature configuration template
  - IEEE 802.1X secret key, in the VPN Interface Ethernet feature configuration template
  - IPsec IKE authentication preshared key, in the VPN Interface IPsec feature configuration template
  - CHAP and PAP passwords, in the VPN Interface PPP Ethernet feature configuration template
  - Wireless LAN WPA key, in the WiFi SSID feature configuration template

Outstanding Issues

The following are outstanding issues in Viptela Software Release 18.1. The number following each issue is the bug number in the Viptela bug-tracking database.

AAA

- The Viptela software does not send a TACACS vendor-specific "service argument" field. [VIP-25629]

Cellular Interfaces

- If you configure IPv6 on a cellular interface, the control connections might go down and come back up continuously. [VIP-21970]

- On a vEdge 100m-NA router, when you configure profile 1 for a wireless WAN, you might see the error "Aborted: 'vpn 0 interface cellular0 profile': Invalid profile 1 : APN missing". [VIP-31721]
- You cannot configure profile 16 in the **interface cellular0 profile** command.

CloudExpress Service

- The CloudExpress vQoE score history value might differ from the score shown for the corresponding application. [VIP-34346]

Configuration and Command-Line Interface

- When you issue the **show vrrp interfaces** command from the vEdge router's CLI, the CLI might not recognize the command and might show a "syntax error: unknown argument" error message. [VIP-23918]
- If a physical interface is part of a bridge, you cannot adjust the MTU on the interface. As a result, the 802.1x interface's MTU has to be lowered to 1496. If the interface needs to also run OSPF, this MTU size can cause an MTU mismatch with other interfaces that have an MTU of 1500. [VIP-26759]
- The **traceroute** command might not work if you specify an IPv6 address for the host. [VIP-30833]
- When two routes exist to the same neighbor, if you specify a single IP address in the **show ip routes** command, the command might return only one of the routes, but if you specify an IPv4 prefix and prefix length, the command returns both routes. [VIP-32736]
- With the **ping source** *ip-address* command, if you type it as **ping so** *ip-address* , the CLI does not autocomplete "so" and the command fails. You must type out the keyword **source** . [VIP-36087]
- In the same sequence in a data policy that you configure on a vManage server, you might not be able to configure both individual ports and port ranges. [VIP-36864]

Forwarding

- For IEEE 802.1X, you cannot configure a RADIUS server for MAC authentication bypass (MAB). [VIP-18492]
- In application-aware routing policy, the salesforce_chatter, oracle_rac, and google_photos applications might not be classified properly. [VIP-21866]
- When you configure a weight on a TLOC that is also being used as a split tunnel, the weight is not used for weighted ECMP across the NATs. [VIP-27534]
- When you switch data traffic from one tunnel to another (for example, from a biz-ethernet to an lte tunnel), a small amount of traffic might be lost. [VIP-27992]
- For a source and destination NAT, return traffic might not be able to reach the VPN that originates the session. [VIP-31299]
- When you configure **policy cloud-qos** on a vEdge Cloud router, a TLOC from the remote site might go down and then come back up when multiple traffic flows are present on the TLOC. [VIP-32369]
- When you configure inbound and outbound port mirroring on the same interface, traffic might be mirrored only in one direction. [VIP-33247]
- If you enable TCP optimization on a vEdge 1000 router, the router might drop ARP responses. [VIP-33507]
- When multiple NAT interfaces are present in VPN 0, port forwarding might not work. [VIP-34086]
- If you disable deep packet inspection (DPI) on a vEdge router, traffic directed towards queue 0 (LLQ) might become bursty or might be dropped. [VIP-34211]
- The TCP optimization process might consume a large amount of CPU even though TCP optimization is not configured. [VIP-36675]
- When you use the **show ip route** command to query a route that is not present in the route table, the command might return no output or no failure message. [VIP-36725]

Interfaces

- When a vEdge VRRP primary is connected to a Cisco switch, the switch might report error messages indicating that the source MAC address is invalid. [VIP-28922]
- When a VRRP backup vEdge router that has been promoted to a primary again becomes a backup, other devices continue to point to the MAC address for the backup router, and traffic is discarded until ARP cache on the other devices expires and is updated with the correct MAC address of primary vEdge, a process that typically takes a few minutes. [VIP-33722]

Policy

- QoS shaping rates might be inaccurate for rates less than 2 Mbps. [VIP-3860]
- A centralized policy that is pushed from the vSmart controller to the vEdge routers might not be applied on the routers. If this occurs, modify the policy and push it again. [VIP-27046]
- On vEdge routers, the **show policy access-list-counters** command might not display any values in the Bytes column. [VIP-28890]
- In vManage NMS, when you use the policy configuration wizard to create policies for a mesh topology, you might need to create an additional policy using a CLI template for the mesh policy to work. This situation is known to occur in a network that has two regions, where each region is mesh that is a subset of the entire network, where each region has its own data center, and where the branch vEdge routers in one region communicate with branch routers in the other region through the data centers. We will call these Region 1 and Region 2. Assume that Region 1 has a control policy that advertises its TLOCs to the data center in Region 2, and Region 2 has a control policy that prevents the spokes and data center in Region 2 from advertising TLOCs to the spokes in Region 1. The result is that the data center in Region 2 repeatedly attempts to form control tunnels to the data center in Region 1, but these attempts fail. As a workaround, you must a policy using a CLI template that allows the data center in Region 2 to exchange TLOCs with the data center in Region 1 and then attach that policy to the vEdge routers. [VIP-29933]
- The vSmart controller might not push a policy to the vEdge routers. [VIP-33016]
- When you issue the **request admin-tech** command, the Forwarding Policy Manager process (fpmd) might crash. [VIP-34031]
- After you change a policy on the vSmart controller, the OMP process (ompd) process might fail and the vSmart controller might crash. [VIP-34098]
- When the vSmart controller pushes a policy to a vEdge router and the push fails, no alarm or trap records the failure. [VIP-34131]

Routing Protocols

- When the OSPF external distance is set to 254, an IP prefix learned first from OMP and then from OSPF as an type E2 route, the route might be redistributed into OMP. [VIP-20542]
- When you are upgrading vEdge routers to Release 16.2.12, the BGP process (bgpd) might crash during the reboot process, when the router is shutting down. [VIP-29523]
- On a vEdge 1000 router, the OSPF process (ospfd) might fail and cause the router to crash. [VIP-30239]
- When you use the iPerf speed tool at 200 Mbps on a vEdge router, the control plane and OSPF neighbors might go down. [VIP-36152]
- The output of the **show bgp summary** and **show bgp neighbors** commands might be incorrect. [VIP-36806]

Security

- When you configure IPsec parameters for data plane security or during an IPsec rekeying operation, you might see a spike in CPU usage on the vEdge router, especially a hardware router. [VIP-31635]

SNMP

- When traffic exceeds 85% of the bandwidth configured on a transport interface, SNMP traps might not get triggered. [VIP-33435]
- The snmpwalk operation might not return SNMP data for the interface ifname. [VIP-35873]

System

- vBond orchestrators might report a large number of control-connection-auth-fail events. [VIP-22976]
- The vManage server might not process events received from vEdge routers. [VIP-28673]
- When a certificate for controllers is about to expire, no syslog message is generated. [VIP-28960]
- When NAT is configured between in a service-side VPN, a ping operation between a vEdge router in that service VPN and another vEdge router reachable through the transport network might be successful even though it should be blocked because of the NAT. [VIP-31078]
- When a vEdge router is unable to reach one of the controllers in a controller, it might not try to reach other controllers in the same group. [VIP-31881]
- On vEdge routers, when you issue an **nping** command for IPv6, the command might fail, and a core file might be created on the router. From vManage NMS, you issue this command from the Monitor ► Network ► Troubleshooting ► Ping pane. From the CLI, you use the **tools nping** command, specifying **options "--ipv6"** . [VIP-31924]
- A vEdge router might choose to establish its control connection to the vManage NMS using an interface on which a tunnel interface is not configured even though an interface with a tunnel interface is operational. [VIP-32011]
- The vdebug log file might contain no entries. [VIP-33662]
- A vSmart controller might crash and create the core file /rootfs.rw/var/crash/core.vtracker.vSmart, indicating an issue with the vtracker process, which pings the vBond orchestrator every second. [VIP-33719]
- When the vManage NMS experiences a kernel panic and reboots, the /var/crash/crash.dump file might be deleted. [VIP-34248]
- You cannot disable ICMP redirect messages. [VIP-36594]

vEdge Hardware

- On a vEdge 100m router, after you execute the **request software reset** command, the router might reboot continuously. [VIP-24149]
- A vEdge 2000 router physical interface might drop packets larger than 1480 bytes that are sent on loopback interfaces. [VIP-27216]
- On a vEdge100m router, the output of the **show interface** command might show the same interface in two different VPNs. [VIP-29069]

vManage NMS

- If you try to configure a vEdge router using vManage configuration templates, you might see errors related to lock-denied problems. As a workaround, reboot the router. [VIP-23826]
- If you use the CLI to modify the organization name, this change might not be reflected on the vManage screens. [VIP-24343]
- When the majority of vManage cluster members are down, you can make changes to the device configuration templates on one of the cluster members that is up, and you can then push these changes when the cluster members come back up. This might lead to a situation in which the configuration templates on the vManage NMSs in the cluster are out of sync. [VIP-26016]
- A vManage NMS might not be able to synchronize its configuration with a vSmart controller. [VIP-26270]
- The vManage server might not process events received from vEdge routers. [VIP-28312]
- When you use the vManage NMS and the CLI **show system status** command, the reboot reason is incorrect; it is shown as unknown. Looking in the /var/log/tmplog/vdebug logs shows that the system reboot happened because of a user-initiated upgrade to Release 17.1.3. [VIP-31222]
- In the vManage AAA feature template, you might not be able to enter the RADIUS secret key even though you can enter that same key in the CLI. [VIP-31856]
- When you push a policy that contains an error to the vSmart controller, the error message might not correctly indicated the cause of the error. [VIP-32253]
- You might not be able to push a configuration template to a vEdge router. [VIP-32277]

- When you are using a vManage cluster, pushing policies to vSmart controllers might time out. [VIP-32630]
- Pushing a configuration template to a vEdge router might time out if the configuration has only one interface and that interface is configured as a last-resort interface. [VIP-33157]
- In a vManage cluster, if you reboot one vManage server from another vManage server (from the Maintenance ► Reboot Devices screen), the vManage configuration database might become out of sync with the actual device configurations. It is recommended that you not reboot a vManage server in this way. [VIP-33625]
- When you copy the configuration database from the primary vManage NMS to bring up a secondary vManage NMS, the certificates for vEdge Cloud routers are not included, and the control plane and data plane for these routers do not come up. [VIP-34085]
- After a vManage NMS silently reboots, it might be out of sync with the vManage cluster. [VIP-35891]
- In Viptela Software Release 17.1.4, when you use OpenStack Heat to create a vManage NMS, the /dev/vdb disk is recognized and mounts. However, in Release 17.2.3, the disk volume might not be automatically recognized and mounted. [VIP-35907]
- In a vManage cluster, when you try to display the Maintenance ► Software Upgrade ► vEdge screen, the screen might not display, and the vManage-Server.Log shows an exception error. [VIP-35926]
- A standalone vManage NMS deployed on ESXi might become inaccessible because the server_config.json file gets corrupted. [VIP-36282]
- When a vBond orchestrator is unreachable or has wrong credentials configured, pushing a vEdge list to it fails with the message “File /home/najmadmin/vedge_serial_numbers must be in home directory”, which does not provide any useful information to the user to understand what is wrong. [VIP-36285]
- When you upgrade the vManage server from Release 17.1.4 to Release 17.2, you might see certificate null pointer exceptions. [VIP-34901]
- The process of installing a new certificate on the vManage NMS might take a long time, eventually failing with the message "Failed to finish the task". However, the output of the **show control valid-vsmarts** command indicates that the certificate installation has succeeded. But because the vManage server is not aware that the task has completed, you are unable to attach configuration templates to the vManage server. [VIP-36579]
- In the vManage Configuration ► Policies ► Centralized Data Policies screen, a user who does not have policy write permission might see the copy, edit, and delete actions in the More Actions icon to the right of a policy listed in the policy table. [VIP-36770]
- The default VPN 512 management feature template is named "Transport VPN", which is confusing because VPN 0 is the transport VPN. [VIP-36771]
- In a vManage cluster running Release 17.2.3, one of the vManage servers might not be able to connect to the configuration database. As a workaround, issue the **request nms all restart** command on the vManage server to restart all NMS services. [VIP-36805]

Fixed Issues

Issues Fixed in Release 18.1.1

Configuration and Command-Line Interface

- When you push a large policy from the vManage NMS to a vSmart controller, it might take a long time for the policy to take effect. [VIP-22115: This issue has been resolved.]

Forwarding

- When a last-resort interface has been initiated and connections on that interface are being brought up, the value of the last-resort hold-down timer might be shown incorrectly in syslog files. [VIP-30423: This issue has been resolved.]

Interfaces

- On a vEdge router WLAN interface, when you configure the RADIUS source interface as loopback0, RADIUS requests might be sourced from random interfaces. [VIP-24240: This issue has been resolved.]

PPPoE

- When you are using PPPoE on a vEdge100m router, when you initially connect to the PPP WAN interface, the interface receives an IP address. But when you unplug the PPP interface and then plug it back in, we do not get an IP address. As a workaround, either reboot the DSL modem or reset the interface from the router's CLI. [VIP-23332: This issue has been resolved.]

Routing Protocols

- When you have configured a BGP peering session to restart after receiving a more than a set number of prefixes from its neighbor, the session might not restart when the number of prefixes is exceeded. [VIP-33780: This issue has been resolved.]

Security

- If an IPv6 address for the IPsec tunnel source interface, the IPsec tunnel does not come up. [VIP-29912: This issue has been resolved.]
- After you upgrade to Release 17.2.3, the vBond orchestrator might take a long time to stabilize and to establish connections to all the vSmart controllers. [VIP-35514: This issue has been resolved.]

SNMP

- When a vEdge 100 reboots, it might start SNMP on the wrong port. [VIP-24700: This issue has been resolved.]

System

- The output of the traceroute command on a vEdge router might be incorrect. [VIP-23072: This issue has been resolved.]
- If you do not use ZTP and if the Organization Name is not set on the vManage NMS and vEdge routers, control connections between the two devices might come up. [VIP-24246: This issue has been resolved.]
- When you connect to the management interface of Viptela devices via SSH, you are placed into a limited shell. However, if you issue the **vshell** command, all operations are executed as the root user. [VIP-32713: This issue has been resolved.]
- On a vEdge 2000 router, BFD sessions may go down and then come back up every 10-60 minutes. As a workaround, disable cflowd and DPI. [VIP-33784: This issue has been resolved.]
- When you access the vManage server from the Mozilla Firefox browser, if you update the interface in area 0 in the OSPF feature template, you might not be able to save the template. As a workaround, use the Chrome browser. [VIP-35584: This issue has been resolved.]

vEdge Hardware

- When a vEdge 2000 router reboots, the reboot reason field might show only a value of 0. [VIP-23941: This issue has been resolved.]

vManage NMS

- In the vManage Dashboard Application-Aware Routing pane, the search engine might not work properly. [VIP-23186: This issue has been resolved.]
- When you try to apply a BGP template to a device, you might see the error "invalid value for: shutdown". [VIP-23258: This issue has been resolved.]
- When you attach a vEdge policy template that contains variables to a device template, during the CSV import action the policy variables might not be populated with the values from the CSV file. As a workaround, manually set the policy values and then import the CSV file. [VIP-23862: This issue has been resolved.]
- On vManage NMS, when you display interface queue statistics in real time, statistics for only one of the eight possible queues might be displayed. [VIP-23898: This issue has been resolved.]
- vManage charts might not properly format values between 0 and 1. [VIP-23910: This issue has been resolved.]

- You might not be able to edit the AAA feature configuration template. [VIP-23970: This issue has been resolved.]
- When an admin user edits an existing CLI template, its device type is sometimes Null, so the user is unable to select the device type. [VIP-24182: This issue has been resolved.]
- A vManage serve might continue to attempt to fetch certificates even though all certificates are installed. [VIP-27416: This issue has been resolved.]
- On vManage NMS, you might not be able to edit a primary configuration template. [VIP-28689: This issue has been resolved.]
- The /client/activity/summary REST call might time out. [VIP-28737: This issue has been resolved.]
- From a vManage server, you might not be able to SSH into a vEdge router that is in staging mode. [VIP-33119: This issue has been resolved.]
- In the vManage Monitor ► Network ► Real Time screen, the output of the Interface Queue Stats command might show information for queue 0 only, showing no information about queues 1 through 7. [VIP-33508: This issue has been resolved.]
- When you query deviceCategory=controllers from the vManage API, the following error might be returned: "%20 characters are spaces and need to be removed". [VIP-33758: This issue has been resolved.]

Issues Fixed in Release 18.1.0

Viptela Software Release 18.1.0 was not released.

YANG Files for Netconf and Enterprise MIB Files

Netconf uses YANG files to install, manipulate, and delete device configurations, and Viptela supports a number of enterprise MIBs. Both are provided in a single tar file. Click the filename below to download the file.

- **YANG and Enterprise MIB files for Release 18.1.1**

Using the Product Documentation

The Viptela product documentation is organized into seven modules:

Module	Description
Getting Started	Release notes for Viptela software releases, information on bringing up the Viptela overlay network for the first time, quick starts for vEdge routers, software download and installation, and an overview of the Viptela solution.
vEdge Routers	How to install, maintain, and troubleshoot vEdge routers and their components. Provides hardware server recommendations for the controller devices—vManage NMS, vSmart controller, and vBond orchestrator servers.
Software Features	Overview and configuration information for software features, organized by software release.
vManage How-Tos	Short step-by-step articles on how to configure, monitor, maintain, and troubleshoot Viptela devices using the vManage NMS.
Command Reference	Reference pages for CLI commands used to configure, monitor, and manage the Viptela devices. Includes reference pages for Viptela software REST API, a programmatic interface for controlling, configuring, and monitoring the Viptela devices in an overlay network.
vManage Help	Help pages for the vManage screens. These pages are also accessible from the vManage GUI.

Tips

- To create a PDF of an article or a guide, click the PDF icon located at the top of the left navigation bar.
- To find information related to an article, see the Additional Information section at the end of each article.
- To help us improve the documentation, click the Feedback button located in the upper right corner of each article page and submit your comments.

Using the Search Engine

- To search for information in the documentation, use the TechLibrary Search box located at the top of each page.
- On the Help results page, you can narrow down your search by selecting the appropriate documentation module at the top of the page. If, for example, you are searching for power supply information for your vEdge router model, select the Hardware module and then select your vEdge router model.
- When a search returns multiple entries with the same title, check the URL to select the article for your hardware platform or software release.
- When the search string is a phrase, the search engine prioritizes the individual words in a phrase before returning results for the entire phrase. For example, the search phrase *full-cone NAT* places links to "NAT" at the top of the search results. If such a search request does not return relevant results, enclose the entire search string in quotation marks (here, for example, *"full-cone NAT"*).

Issues

- The maximum PDF page limit is 50 pages.
- It is recommended that you use the Chrome browser when reading the production documentation. Some of the page elements, such as the PDF icon, might not display properly in Safari.
- The screenshots for the vManage NMS screens that are included in the vManage help files and other documentation articles might not match the vManage NMS software screens. We apologize for the inconvenience.

Requesting Technical Support

To request technical support, send email to support@viptela.com .

To provide documentation feedback or comments, send email to docs@viptela.com .

Revision History

Release 18.1.0 was not released.
Revision 1—Release 18.1.1, March 30, 2018

System and Interfaces

System and Interfaces Overview

Setting up the basic system-wide functionality of Viptela network devices is a simple and straightforward process. These basic parameters include defining host properties, such as name and IP address; setting time properties, including NTP; setting up user access to the devices; defining system log (syslog) parameters; and creating network interfaces. In addition, the Viptela software provides a number of management interfaces for accessing the Viptela devices in the overlay network.

Host Properties

All Viptela devices have basic system-wide properties that specify information that the Viptela software uses to construct a view of the network topology. Each device has a system IP address, which provides a fixed location of the device in the overlay network. This address, whose function is similar to that of a router ID on a router, is independent of any of the interfaces and interface addresses on the device. The system IP address is a component of each device's TLOC address.

A second host property that must be set on all Viptela devices is the IP address of the vBond orchestrator for the network domain, or a DNS name that resolves to one or more IP addresses for vBond orchestrators. As discussed in **Components of the Viptela Solution** , the vBond orchestrator automatically orchestrates the bringup of the overlay network, providing the introductions that allow vEdge routers and vSmart controllers to locate each other.

Two other system-wide host properties are required on all devices, except for the vBond orchestrators, to allow the Viptela software to construct a view of the topology: the domain identifier and the site identifier.

To configure the host properties, see **Viptela Overlay Network Bringup** .

Time and NTP

The Viptela software implements the Network Time Protocol (NTP) to synchronize and coordinate time distribution across the Viptela overlay network. NTP uses a returnable-time design in which a distributed subnet of time servers operating in a self-organizing, hierarchical primary-subordinate configuration synchronizes local clocks within the subnet to national time standards by means of wire or radio. The servers also can redistribute reference time using local routing algorithms and time daemons. NTP is defined in RFC 5905, Network Time Protocol Version 4: Protocol and Algorithms Specification .

User Authentication and Access with AAA, RADIUS, and TACACS+

The Viptela software uses Authentication, Authorization, and Accounting (AAA) to provide security for Viptela devices on the network. AAA, in combination with RADIUS and TACACS+ user authentication, controls which users are allowed access to Viptela devices and what operations they are authorized to perform once they are logged in or connected to the devices.

Authentication refers to the process by which the user trying to access the device is authenticated. To access Viptela devices, users log in with either a standard or a custom username and a password. The local device can authenticate users, or authentication can be performed by a remote device, either by a Remote Authentication Dial-In User Service (RADIUS) server or by a Terminal Access Controller Access-Control System (TACACS+) system, or by both in sequence.

Authorization determines whether the user is authorized to perform a given activity on the Viptela device. In the Viptela software, authorization is implemented using role-based access. Access is based on groups that are configured on the Viptela devices. A user can be a member of one or more groups. External groups are also considered when performing authorization; that is, the Viptela software retrieves group names from RADIUS or TACACS+ servers. Each group is assigned privileges that authorize the group members to perform specific functions on the Viptela device. These privileges correspond to specific hierarchies of the configuration commands and the corresponding hierarchies of operational commands that members of the group are allowed to view or modify.

The Viptela software does not implement AAA accounting.

For more information, see Role-Based Access with AAA .

Authentication for WANs and WLANs

For wired networks (WANs), vEdge routers can run IEEE 802.1X software to prevent unauthorized network devices from gaining access to the WAN. IEEE 802.1X is a port-based network access control (PNAC) protocol that uses a client–server mechanism to provide authentication for devices wishing to connect to the network. You enable 802.1X on vEdge router interfaces to have the router act as an 802.1X authenticator, responsible for authorizing or denying access to network devices.

IEEE 802.1X authentication requires three components:

- Supplicant—Client device, such as a laptop, that requests access to the WAN. In the Viptela overlay network, a supplicant is any service-side device that is running 802.1X-compliant software. These devices send network access requests to the vEdge router.
- Authenticator—Network device, here a vEdge router, that provides a barrier to the WAN. In the overlay network, you can configure an interface vEdge router to act as an 802.1X authenticator. The vEdge router supports both controlled and uncontrolled ports. For controlled ports, the router acts as an 802.1X port access entity (PAE), allowing authorized network traffic and preventing unauthorized network traffic ingressing to and egressing from the controlled port. For uncontrolled ports, the router, acting as an 802.1X PAE, transmits and receives Extensible Authentication Protocol over IEEE 802 (EAP over LAN, or EAPOL) frames.
- Authentication server—Host running authentication software that validates and authenticates supplicants that want to connect to the WAN. In the overlay network, this host is an external RADIUS server. This RADIUS server authenticates each client connected to the vEdge router's 802.1X port interface and assigns the interface to a VLAN before the client is allowed to access any of the services offered by the router or by the LAN.

For wireless LANs (WLANs), vEdge routers can run IEEE 802.11i prevents unauthorized network devices from gaining access to the WLANs. IEEE 802.11i implements Wi-Fi Protected Access (WPA) and Wi-Fi Protected Access II (WPA2) to provide authentication and encryption for devices that want to connect to a WLAN. WPA authenticates individual users on the WLAN using a username and password. WPA uses the Temporal Key Integrity Protocol (TKIP), which is based on the RC4 cipher. WPA2 implements the NIST FIPS 140-2–compliant AES encryption algorithm along with IEEE 802.1X-based authentication, to enhance user access security over WPA. WPA2 uses the Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP), which is based on the AES cipher. Authentication is done either using preshared keys or through RADIUS authentication.

Network Interfaces

In the Viptela overlay network design, interfaces are associated with VPNs. The interfaces that participate in a VPN are configured and enabled in that VPN. Each interface can be present only in a single VPN.

The overlay network has three broad types of VPNs:

- Transport VPN—This is VPN 0. All interfaces activated in this VPN connect to a transport network of some type, such as the Internet, metro Ethernet, or an MPLS cloud, and these interfaces carry overlay network control traffic. The interfaces in VPN 0 are referred to as transport-side interfaces. You can configure transport VPNs on all Viptela devices.
- Service VPNs—These are all VPNs up through VPN 65535 except for VPN 0 and VPN 512. You can configure service-side VPNs only on vEdge routers. All service-side interfaces activated in these VPNs connect to a local or branch network that is generally located at the same site as the vEdge router. These interfaces carry data traffic throughout the overlay network.
- Management VPN—VPN 512 handles out-of-band management traffic. You can configure the management VPN on all Viptela devices.

For all Viptela devices, you can configure transport interfaces to run either IPv4 or IPv6, or you can configure them to run both, for a dual-stack implementation.

For each network interface, you can configure a number of interface-specific properties, such as DHCP clients and servers, VRRP, interface MTU and speed, and PPPoE. At a high level, for an interface to be operational, you must configure an IP address for the interface and mark it as operational (no shutdown). In practice, you always configure additional parameters for each interface.

Management Interfaces

Management interfaces provide access to devices in Viptela overlay network, allowing you to collect information from the devices in an out-of-band fashion and to perform operations on the devices, such as configuring and rebooting them.

The following management interfaces are available:

- Command-line interface (CLI)
- IP Flow Information Export (IPFIX)
- RESTful API
- SNMP
- System logging (syslog) messages
- vManage web server

CLI

You can access a command-line interface (CLI) on each Viptela device, and from the CLI you configure overlay network features on the local device and gather operational status and information regarding that device. While a CLI is available, it is strongly recommended that you configure and monitor all Viptela network devices from a vManage web server, which provides visual views of network-wide operations and device status, including drill-downs that display details operation and status data. In addition, the vManage web server provides straightforward tools for bringing up and configuring overlay network devices, including bulk operations for setting up multiple devices simultaneously.

You access the CLI either by establishing an SSH session to a Viptela device. For a hardware vEdge router, you can also connect to the device's console port.

For a Viptela device that is being managed by a vManage NMS, if you create or modify the configuration from the CLI, those changes are overwritten by the configuration that is stored in the vManage configuration database.

From a device's CLI, you can log in to the underlying shell running on the device, referred to as the vshell. The vshell filesystem stores system logging (syslog) and other files that contain status information about the device. The vManage NMS periodically retrieves the information in these files and makes the information available to be displayed on a vManage web server.

It is recommended that you log in to a device's shell only when you are working to debug an issue with the device or with the network.

IPFIX

the IP Flow Information Export (IPFIX) protocol, also called cflowd, is a tool for monitoring the traffic flowing through vEdge routers in the overlay network and exporting information about the traffic to a flow collector. The exported information is sent in template reports, which contain both information about the flow and data extracted from the IP headers of the packets in the flow.

The Viptela cflowd performs 1:1 traffic sampling. Information about all flows is aggregated in the cflowd records; flows are not sampled. vEdge routers do not cache any of the records that are exported to a collector.

The Viptela cflodw software implements cflowd version 10, as specified in RFC 7011 and RFC 7012 .

For a list of elements exported by IPFIX, see Traffic Flow Monitoring with Cflowd .

To enable the collection of traffic flow information, you create data policies that identify the traffic of interest and then direct that traffic to a cflowd collector. For more information, see Traffic Flow Monitoring with Cflowd.

You can also enable cflowd visibility directly on vEdge routers without configuring data policy so that you can perform traffic flow monitoring on traffic coming to the router from all VPNs in the LAN. You then monitor the traffic from the vManage GUI or from the router's CLI.

RESTful API

The Viptela software provides a RESTful API, which is a programmatic interface for controlling, configuring, and monitoring the Viptela devices in an overlay network. You access the RESTful API through the vManage web server.

The Viptela RESTful API calls expose the functionality of Viptela software and hardware features and of the normal operations you perform to maintain Viptela devices and the overlay network itself.

For more information, see **Viptela REST APIs** .

SNMP

The Simple Network Management Protocol (SNMP) allows you to manage all Viptela devices in the overlay network. The Viptela software supports SNMP Version 1, Version 2 (also known as Version 2c, or v2c), and SNMPv3. All three versions of SNMP are supported simultaneously. For SNMPv1 and SNMPv2, the Viptela software does not include any of the security features that were originally included in the IETF SNMP drafts, but were later dropped because of the inability to standardize on a particular method. In SNMP v1, SNMP v2 user access is controlled by communities, so AAA rules for users and user groups are not enforced in this case.

You can configure basic SNMP properties—device name, location, contact, and community—that allow the device to be monitored by an SNMP NMS.

You can configure trap groups and SNMP servers to receive traps.

The object identifier (OID) for the Internet port of the SNMP MIB is 1.3.6.1. The OID for the private portion of the Viptela MIB is 1.3.6.1.4.1.41916.

For a list of supported MIBs, see Supported SNMP MIBs .

SNMP traps are asynchronous notifications that a Viptela device sends to an SNMP management server. Traps notify the management server of events, whether normal or significant, that occur on the Viptela device. By default, SNMP traps are not sent to an SNMP server. Note that for SNMPv3, the PDU type for notifications ie either SNMPv2c inform (InformRequest-PDU) or trap (Trapv2-PDU). For a list of supported trap types, see Configuring SNMP .

Syslog Messages

System logging operations use a mechanism similar to the UNIX syslog command to record system-wide, high-level operations that occur on the Viptela devices in the overlay network. The log levels (priorities) of the messages are the same as those in standard UNIX commands, and you can configure which priority of syslog messages are logged. Messages can be logged to a file on the Viptela device or to a remote host.

vManage NMS

The vManage NMS is a centralized network management system that allows configuration and management of all Viptela devices in the overlay network and provides a dashboard into the operations of the entire network and of individual devices in the network. Each vManage NMS runs on a web server in the network. Three or more vManage web servers are consolidated into a vManage cluster to provide scalability and management support for up to 6,000 vEdge routers, to distribute vManage functions across multiple devices, and to provide redundancy of network management operations.

Additional Information

Configuring SNMP
Configuring Time and Location
Configuring User Access and Authentication

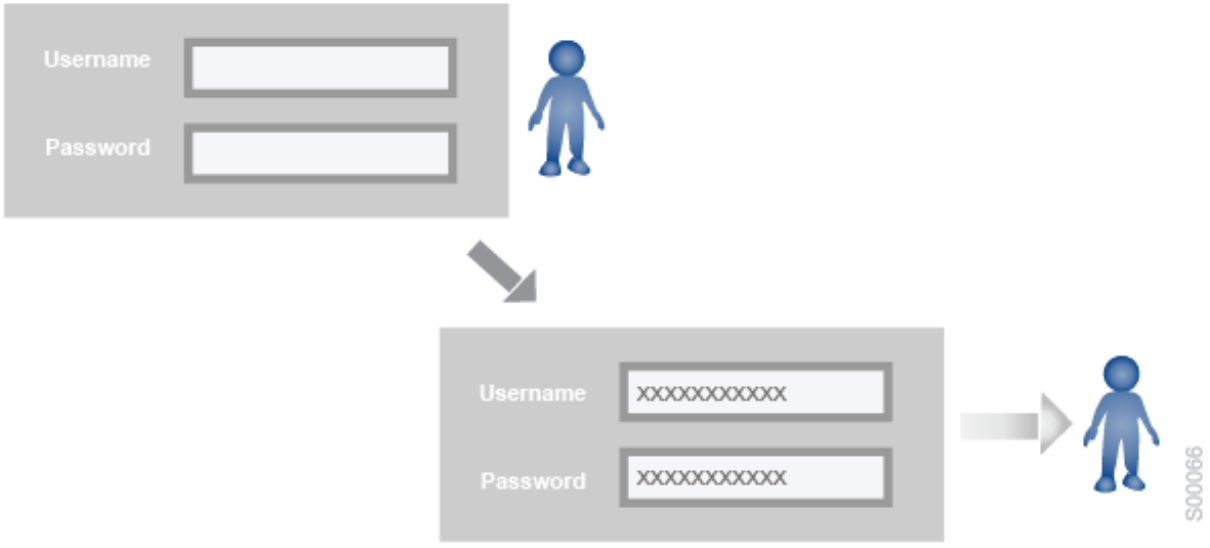
Role-Based Access with AAA

The Viptela AAA software implements role-based access to control the authorization permissions for users on Viptela devices. Role-based access consists of three components:

- Users are those who are allowed to log in to a Viptela device.
- User groups are collections of users.
- Privileges are associated with each group. They define the commands that the group's users are authorized to issue.

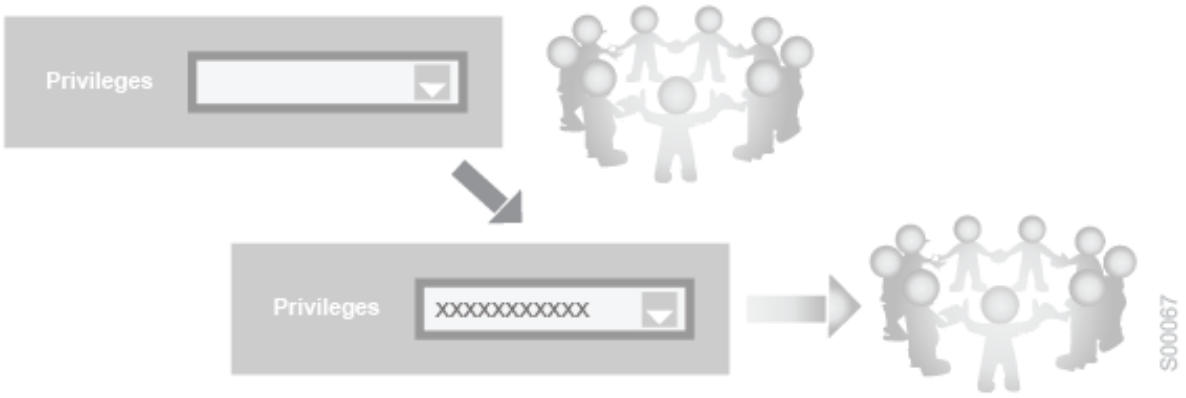
Users and User Groups

All users who are permitted to perform operations on a Viptela device must have a login account. For the login account, you configure a username and a password on the device itself. These allow the user to log in to that device. A username and password must be configured on each device that a user is allowed to access.

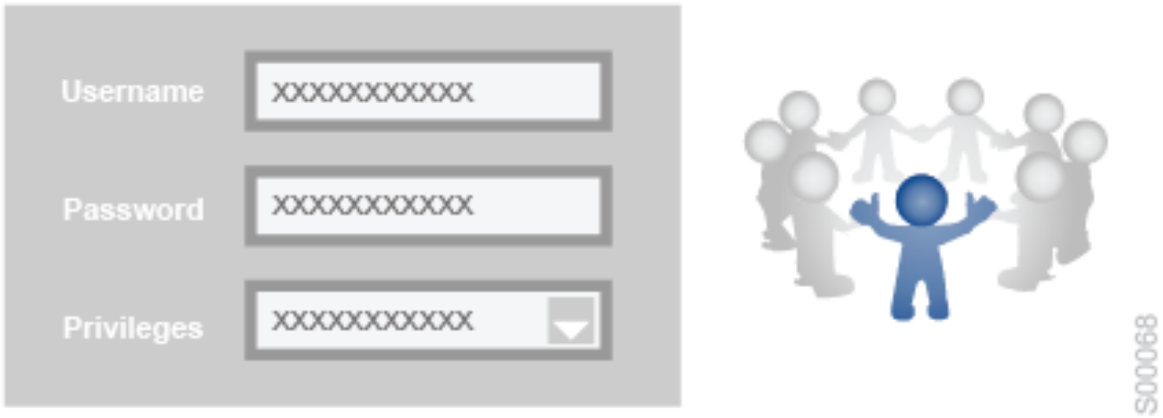


The Viptela software provides one standard username, **admin** , which is a user who has full administrative privileges, similar to a UNIX superuser. By default, the **admin** username password is **admin** . You cannot delete or modify this username, but you can and should change the default password.

User groups pool together users who have common roles, or privileges, on the Viptela device. As part of configuring the login account information, you specify which user group or groups that user is a member of. You do not need to specify a group for the **admin** user, because this user is automatically in the user group **netadmin** and is permitted to perform all operations on the Viptela device.



The user group itself is where you configure the privileges associated with that group. These privileges correspond to the specific commands that the user is permitted to execute, effectively defining the role-based access to the Viptela software elements.



The Viptela software provides three standard user groups. The two groups **basic** and **operator** are configurable. While you can use these two groups for any users and privilege levels, the **basic** group is designed to include users who have permission to both view and modify information on the device, while the **operator** group is designed to include users who have permission only to view information. The third group is **net admin** , which is non-configurable. By default, it includes the **admin** user. You can add other users to this group. Users in this group are permitted to perform all operations on the device.

Privileges for Role-Based Access

Role-based access privileges are arranged into five categories, which are called *tasks:*

- Interface—Privileges for controlling the interfaces on the Viptela device.
- Policy—Privileges for controlling control plane policy, OMP, and data plane policy.
- Routing—Privileges for controlling the routing protocols, including BFD, BGP, OMP, and OSPF.
- Security—Privileges for controlling the security of the device, including installing software and certificates. Only users belonging to the **netadmin** group can install software on the system.

- System—General systemwide privileges.

The tables in the following sections detail the AAA authorization rules for users and user groups. These authorization rules apply to commands issued from the CLI and to those issued from Netconf.

User Authorization Rules for Operational Commands

The user authorization rules for operational commands are based simply on the username. Any user who is allowed to log in to the Viptela device can execute most operational commands. However, only the **admin** user can issue commands that affect the fundamental operation of the device, such as installing and upgrading the software and shutting down the device.

Note that any user can issue the **config** command to enter configuration mode, and once in configuration mode, they are allowed to issue any general configuration command. Also, any user is allowed to configure their password by issuing the **system aaa user** *self* **password** *password* command and then committing that configuration change. For the actual commands that configure device operation, authorization is defined according to user group membership. See User Group Authorization Rules for Configuration Commands .

The following tables lists the AAA authorization rules for general CLI commands. All the commands are operational commands except as noted. Also, some commands available to the "admin" user are available only if that user is in the "netadmin" user group.

CLI Command	Any User	Admin User
clear history	X	X
commit confirm	X	X
complete-on-space	X	X
config	X	X
exit	X	X
file	X	X
help	X	X
[no] history	X	X
idle-timeout	X	X
job	X	X
logout	—	X (users in netadmin group only)
monitor	X	X
nslookup	X	X
paginate	X	X
ping	X	X
poweroff	—	X (users in netadmin group only)
prompt1	X	X
prompt2	X	X
quit	X	X

reboot	—	X (users in netadmin group only)
request aaa request admin-tech request firmware request interface-reset request nms request reset request software	—	X (users in netadmin group only)
request execute request download request upload	X	X
*request (everything else)	—	X
rollback (configuration mode command)	—	X (users in netadmin group only)
screen-length	X	X
screen-width	X	X
show cli	X	X
show configuration commit list	X	X
show history	X	X
show jobs	X	X
show parser dump	X	X
show running-config	X	X
show users	X	X
system aaa user *self* password *password* (configuration mode command) (Note: A user cannot delete themselves)		
tcpdump	X	X
timestamp	X	X
tools ip-route	X	X
tools netstat	X	X
tools nping	X	X
traceroute	X	X
vshell	X	X (users in netadmin group only)

User Group Authorization Rules for Operational Commands

The following table lists the user group authorization roles for operational commands.

Operational Command	Interface	Policy	Routing	Security	System
clear app		X			
clear app-route		X			
clear arp	X				
clear bfd			X		X
clear bgp			X		X
clear bridge	X				
clear cellular	X				
clear control				X	
clear crash					X
clear dhcp					X
clear dns					X
clear igmp			X		
clear installed-certificates				X	
clear interface	X				
clear ip			X		
clear notification					X
clear omp			X		
clear orchestrator				X	
clear ospf			X		
clear pim			X		
clear policy		X			
clear pppoe	X				
clear system					X
clear tunnel				X	
clear wlan	X				
clear ztp				X	X
clock					X
debug bgp			X		
debug cellular	X				
debug cflowd		X			
debug chmgr					X

debug config-mgr					X
debug dhcp-client					X
debug dhcp-helper					X
debug dhcp-server					X
debug fpm		X			
debug ftm					X
debug igmp			X		
debug netconf					X
debug omp			X		
debug ospf			X		
debug pim			X		
debug resolver			X		
debug snmp					X
debug sysmgr					X
debug transport					X
debug ttm					X
debug vdaemon				X	X
debug vrrp				X	
debug wlan	X				
request certificate				X	
request control-tunnel				X	
request controller				X	
request controller-upload				X	
request csr				X	
request device				X	
request device-upload				X	
request on-vbond-controller				X	
request port-hop				X	
request root-cert-chain				X	
request security				X	
request vedge				X	

request vedge-upload				X	
request vsmart-upload				X	
show aaa					X
show app		X			
show app-route		X			
show arp	X				
show bfd			X		X
show bgp			X		
show boot-partition					X
show bridge	X				
show cellular	X				
show certificate				X	
show clock					X
show control				X	X
show crash					X
show debugs—same as debug commands					
show dhcp					X
show external-nat				X	X
show hardware					X
show igmp			X		
show interface	X				
show ip			X		X
show ipsec				X	
show licenses					X
show logging					X
show multicast			X		
show nms-server					X
show notification					X
show ntp					X
show omp		X	X		X
show orchestrator				X	

show ospf			X		
show pim			X		
show policer		X			
show policy		X			
show ppp	X				
show pppoe	X				
show reboot					X
show security-info				X	
show software					X
show system					X
show transport					X
show tunnel				X	
show uptime					X
show users					X
show version					X
show vrrp	X				
show wlan	X				
show ztp				X	

User Group Authorization Rules for Configuration Commands

The following table lists the user group authorization rules for configuration commands.

Configuration Command	Interface	Policy	Routing	Security	System
apply-policy		X			
banner					X
bfd			X		X
bridge	X				
omp		X	X		X
policy		X			
security				X	X
snmp					X
system					X

vpn interface	X				
vpn ip			X		
vpn router			X		
vpn service			X		
vpn (everything else, including creating, deleting, and naming)					X
wlan	X				

Additional Information

aaa
show aaa usergroup
Configuring User Access and Authentication

Supported SNMP MIBs

The Viptela software supports the MIBs listed in the table below.

For information about downloading these MIB files, see the **release notes** for your software release.

MIB

Description

Standard MIBs

Interface MIB
(IF-MIB)

Objects related to device interfaces, including the number of interfaces present on the device and a table of interface properties, status, and statistics. The values returned are similar to those in the **show interface** CLI command and the Interface Detail optio in the vManage Monitor ► Network ► Real Time screen.

System MIB
(SNMPv2-MIB)

Systemwide properties of the device, such as description, name, and location. The values returned are similar to those in the **show system status** CLI command and the System Information and System Status options in the vManage Monitor ► Network ► Real Time screen.

Enterprise MIBs

Application-aware routing
(VIPTELA-APP-ROUTE.mib)

Operational data related to application-aware routing. The values returned are similar to those in the **show app-route stats** and **show app-route sla-class** CLI commands, and the App Routes Statistics and App Routes SLA Class options in the vManage Monitor ► Network ► Real Time screen.

BFD
(VIPTELA-BFD.mib)

Operational data related to the Bidirectional Forwarding Protocol (BFD). The values returned are similar to those in the **show bfd history** , **show bfd sessions** , and **show bfd summary** CLI commands, and the BFD History, BFD Sessions, and BFD Summary options in the vManage Monitor ► Network ► Real Time screen.

BGP
(VIPTELA-OPER-BGP.mib)

Operational data related to BGP. The values returned are similar to those in the **show bgp neighbor** , **show bgp routes** , and **show bgp summary** CLI commands, and the BGP Neighbors, BGP Routes, and BGP Summary options in the vManage Monitor ► Network ► Real Time screen.

Bridging
(VIPTELA-BRIDGE.mib)

Operational data related to bridging. The values returned are similar to those in the **show bridge interface** , **show bridge mac** , and **show bridge table** CLI commands, and the Bridge Interface, Bridge MAC, and Bridge Table options in the vManage Monitor ► Network ► Real Time screen.

Hardware
(VIPTELA-HARDWARE.mib)

Operational data related to Viptela hardware devices. The values returned are similar to those in the **show hardware alarms** , **show hardware environment** , **show hardware inventory** , and **show hardware temperature-thresholds** CLI commands, and the Hardware Alarms, Hardware Environment, Hardware Inventory, and Hardware Temperature Thresholds options in the vManage Monitor ► Network ► Real Time screen.

Multicast and PIM
(VIPTELA-OPER-MULTICAST.mib)

Operational data related to PIM and other multicast protocols. The values returned are similar to those in the **show igmp groups** , **show igmp interface** , **show igmp statistics** , **show igmp summary** , **show multicast rpf** , **show multicast topology** , **show multicast replicator** , **show multicast tunnel** , **show pim interface** , **show pim neighbor** , **show pim rp-mapping** , and **show pim statistics** command, and the various IGMP, Multicast, and PIM options in the vManage Monitor ► Network ► Real Time screen.

OMP
(VIPTELA-OMP.mib)

Operational data related to Overlay Management Protocol (OMP). The values returned are similar to those in the **show omp multicast-auto-discover** , **show omp multicast-routes** , **show omp routes** , **show omp services** , **show omp summary** , and **show omp tlocs** CLI commands, and the OMP options in the vManage Monitor ► Network ► Real Time screen.

OSPF
(VIPTELA-OPER-OSPF.mib)

Operational data related to OSPF. The values returned are similar to those in the **show ospf database** , **show ospf database-summary** , **show ospf interface** , **show ospf neighbor** , **show ospf process** , and **show ospf routes** CLI commands, and the various OSPF options in the vManage Monitor ► Network ► Real Time screen.

Policy
(VIPTELA-POLICY.mib)

Operational data related to policy, access lists, and policers. The values returned are similar to those in the **show policers** , **show policy access-list-associations** , **show policy access-list-counters** , **show policy access-list-names** , **show policy access-list-policers** , **show policy app-route-policy-filter** , **show policy data-policy-filter** , **show policy from-vsmart** , **show policy qos-map-info** , **show policy qos-scheduler-info** , and **show policy rewrite-associations** CLI commands, and the Policer Information and various Policy options in the vManage Monitor ► Network ► Real Time screen.

Security
(VIPTELA-SECURITY.mib)

Operational data related to control plane and data plane security. The values returned are similar to those in the **show control affinity status** , **show control connections** , **show control connections-history** , **show control local-properties** , **show control statistics** , **show control summary** , **show control valid-vedges** , **show control valid-vsmarts** , **show ipsec inbound-connections** , **show ipsec local-sa** , **show ipsec outbound-connections** , **show orchestrator connections** , **show orchestrator connections-history** , **show orchestrator local-properties** , **show orchestrator statistics** , **show orchestrator summary** , **show orchestrator valid-vedges** , **show orchestrator valid-**

vsmarts , **show tunnel gre-keepalives** , **show tunnel statistics** , and **show ztp entries** CLI commands, and the Control, IPsec, Tunnel GRE Keepalives, and Tunnel Statistics options in the vManage Monitor ► Network ► Real Time screen.

SNMP traps
(VIPTELA-TRAPS.mib)

SNMP notifications and traps.

System
(VIPTELA-OPER-SYSTEM.mib)

Operational data related to the device's overall system status. The values returned are similar to those in the **show aaa usergroup** , **show crash** , **show ntp associations** , **show ntp peer** , **show reboot history** , **show software** , **show system statistics** , **show system status** , **show transport connection** , and **show users** , and the Crash Log, NTP Associations, NTP Peers, Reboot History, Software Versions, System Information, System Status, Transport Connections, and Users options in the vManage Monitor ► Network ► Real Time screen.

VPNs
(VIPTELA-OPER-VPN.mib)

Operational data related to VPN-specific functions, including cflowd, deep packet inspection (DPI), DHCP, interfaces, routing tables, NAT interfaces, PPPoE and PPP interfaces, and VRRP. The values returned are similar to those in the **show app cflowd collector** , **show app cflowd flows** , **show app cflowd statistics** , **show arp** , **show app dpi applications** , **show app dpi flows** , **show app dpi supported-applications** , **show app dpi summary statistics** , **show dhcp interface** , **show dhcp server** , **show interface** , **show interface sfp detail** , **show interface sfp diagnostic** , **show ip fib** , **show ip mfib oil** , **show ip mfib stats** , **show ip mfib summary** , **show ip nat filter** , **show ip nat interface** , **show ip nat interface-statistics** , **show ip routes** , **show pppoe session** , **show pppoe statistics** , **show ppp interface** , and **show vrrp** CLI commands, and the Cflowd, DPI, DHCP, Interface, IP, NAT, PPP, PPPoE, and VRRP options in the vManage Monitor ► Network ► Real Time screen.

Wireless WANs
(VIPTELA-WWAN.mib)

Operational data related to cellular interfaces. The values returned are similar to those in the **show cellular modem** , **show cellular network** , **show cellular profiles** , **show cellular radio** , **show cellular sessions** , and **show cellular status** CLI commands, and the various Cellular options in the vManage Monitor ► Network ► Real Time screen.

For information about downloading these MIB files, see the **release notes** for your software release.

Additional Information

Configuring SNMP
System and Interfaces Overview

Configuring User Access and Authentication

This article describes how to use AAA in combination with RADIUS and TACACS+ to configure authentication, authorization, and accounting for users wishing to access Viptela devices.

Configuring AAA

AAA allows you to configure local users on the Viptela device. AAA configuration is done in two steps:

- Configure users—First, you configure usernames and passwords for individuals who are allowed to access the Viptela device. The Viptela software provides one standard username, **admin** , and you can also create custom usernames, as needed.
- Configure groups—Second, you place users in groups, which define the specific configuration and operational commands that the users are authorized to view and modify. A single user can be in one or more groups. See Role-Based Access for AAA for more information about user and group privileges and the authorization that they provide.

Creating Users

The Viptela software provides one standard username, **admin** . Only a user who is logged is as the admin user is permitted to create additional users.

To create a user account, configure the username and password, and place the user into a group:

```
Viptela(config)# system aaa
Vipteal(config)# user username password password
Viptela(config-aaa)# group group-name
```

username can be 1 to 128 characters long, and it must start with a letter. The name can contain only lowercase letters, the digits 0 through 9, hyphens (-), underscores (_), and periods (.). The name cannot contain any uppercase letters. Some usernames are reserved, so you cannot configure them. For a list of them, see the **aaa** configuration command.

password is the password for the user. Each username must have a password, and each user is allowed to change their own password. The CLI immediately encrypts the string and never displays a readable version of the password. When a user is logging in to the Viptela device, they have five chances to enter the correct password. After the fifth incorrect attempt, the user is locked out of the device, and they must wait 15 minutes before attempting to log in again.

group-name is the name of one of the standard Viptela groups (**basic** , **netadmin** , or **operator**) or of a group configured with the **usergroup** command (discussed below). If an **admin** user changes the permission of a user by changing their group, and if that user is currently logged in to the device, the user is logged out and must log back in again.

The factory-default password for the **admin** username is **admin** . It is strongly recommended that you modify this password the first time you configure a Viptela device.

```
Viptela(config)# system aaa admin password password
```

Configure the password as an ASCII string. The CLI immediately encrypts the string and never displays a readable version of the password. For example:

```
vEdge(config-user-admin)# show config
system
 aaa
  user admin
   password $1$xULc8yYH$k71cTjvKESmeIGgImNDaC.
  !
  user eve
   password $1$8z3q4qoU$F6DMBr9vPBF0s/sl45ax5.
   group    basic
  !
 !
!
```

If you are using RADIUS to perform AAA authentication, you can configure a specific RADIUS server to use to verify the password:

```
Viptela(config)# system aaa radius-servers tag
```

tag is a string that you defined with the **radius server tag** command, as described below.

Creating Groups

The Viptela software provides three fixed group names: **basic** , **netadmin** , and **operator** . The username **admin** is automatically placed in the **netadmin** usergroup.

To create a custom group with specific authorization, configure the group name and privileges:

```
Viptela(config)# system aaa usergroup group-name task privilege
```

group-name can be 1 to 128 characters long, and it must start with a letter. The name can contain only lowercase letters, the digits 0 through 9, hyphens (-), underscores (_), and periods (.). The name cannot contain any uppercase letters Some group names are reserved, so you cannot configure them. For a list of them, see the aaa configuration command.

If a remote RADIUS or TACACS+ server validates authentication but does not specify a user group, the user is placed into the user group **basic** . If a remote server validates authentication and specifies a user group (say, X) using VSA Viptela-Group-Name, the user is placed into that user group only. However, if that user is also configured locally and belongs to a user group (say, Y), the user is placed into both the groups (X and Y).

In the **task** option, list the privilege roles that the group members have. The role can be one or more of the following: **interface** , **policy** , **routing** , **security** , and **system** .

In the following example, the **basic** user group has full access to the **system** and **interface** portions of the configuration and operational commands, and the **operator** user group can use all operational commands but can make no modifications to the configuration:

```
vEdge# show running-config system aaa
system
 aaa
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  user admin
   password $1$tokPB7tf$vchR2JI9Sw1/dqgkqup9S.
  !
 !
!
```

Configuring RADIUS Authentication

To have a Viptela device use RADIUS servers for user authentication, configure one or up to 8 servers:

```
Viptela(config)#  system radius
Viptela(config-radius)# server ip-address
Viptela(config-server)# secret-key password
Viptela(config-server)# priority number
Viptela(config-server)# auth-port port-number
Viptela(config-server#) acct-port port-number
Viptela(config-server)# source-interface interface-name
Viptela(config-server)# tag tag
Viptela(config-server)# vpn vpn-id
```

For each RADIUS server, you must configure, at a minimum, its IP address and a password, or key. You can specify the key as a clear text string up to 32 characters long or as an AES 128-bit encrypted key. The local device passes the key to the RADIUS server. The password must match the one used on the server. To configure more than one RADIUS server, include the **server** and **secret-key** commands for each server.

The remaining RADIUS configuration parameters are optional.

To set the priority of a RADIUS server, as a means of choosing or load balancing among multiple RADIUS servers, set a priority value for the server. The priority can be a value from 0 through 7. A server with a lower priority number is given priority over one with a higher number.

By default, the Viptela device uses port 1812 for authentication connections to the RADIUS server and port 1813 for accounting connections. To change these port numbers, use the **auth-port** and **acct-port** commands.

If the RADIUS server is reachable via a specific interface, configure that interface with the **source-interface** command.

You can tag RADIUS servers so that a specific server or servers can be used for AAA, IEEE 802.1X, and IEEE 802.11i authentication and accounting. Define the tag here, with a string from 4 to 16 characters long. Then associate the tag with the **radius-servers** command when you configure AAA, and when you configure interfaces for 802.1X and 802.11i.

If the RADIUS server is located in a different VPN from the Viptela device, configure the server's VPN number so that the Viptela device can locate it. If you configure multiple RADIUS servers, they must all be in the same VPN.

When a Viptela device is trying to locate a RADIUS server, it goes through the list of servers three times. To change this, use the **retransmit** command, setting the number to a value from 1 to 1000:

```
Viptela(config-radius)# retransmit number
```

When waiting for a reply from the RADIUS server, a Viptela device waits 3 seconds before retransmitting its request. To change this time interval, use the **timeout** command, setting a value from 1 to 1000 seconds:

```
Viptela(config-radius)# timeout seconds
```

Configuring TACACS+ Authentication

To have a Viptela device use TACACS+ servers for user authentication, configure one or up to 8 servers:

```
Viptela(config)# system tacacs
Viptela(config)# server ip-address
Viptela(config-server)# secret-key password
Viptela(config-server)# priority number
Viptela(config-server)# auth-port port-number
Viptela(config-server)# source-interface interface-name
Viptela(config-server)# vpn vpn-id
```

For each TACACS+ server, you must configure, at a minimum, its IP address and and a password, or key. You can specify the key as a clear-text string up to 32 characters long or as an AES 128-bit encrypted key. The local device passes the key to the TACACS+ server. The password must match the one used on the server. To configure more than one TACACS+ server, include the **server** and **secret-key** commands for each server.

The remaining TACACS+ configuration parameters are optional.

To set the priority of a RADIUS server, as a means of choosing or load balancing among multiple RADIUS servers, set a priority value for the server. The priority can be a value from 0 through 7. A server with a lower priority number is given priority over one with a higher number.

By default, the Viptela device uses port 49 to connect to the TACACS+ server. To change this, use the **auth-port** command.

If the TACACS+ server is reachable via a specific interface, configure that interface with the **source-interface** command.

If the TACACS+ server is located in a different VPN from the Viptela device, configure the server's VPN number so that the Viptela device can locate it. If you configure multiple TACACS+ servers, they must all be in the same VPN.

By default, PAP is used as the authentication type for the password for all TACACS+ servers. You can change the authentication type to ASCII:

```
Viptela(config-tacacs)# authentication ascii
```

When waiting for a reply from the TACACS+ server, a Viptela device waits 5 seconds before retransmitting its request. To change this time interval, use the **timeout** command, setting a value from 1 to 1000 seconds:

```
Viptela(config-tacacs)# timeout seconds
```

Configuring the Authentication Order

The authentication order dictates the order in which authentication methods are tried when verifying user access to a Viptela device through an SSH session or a console port. The default authentication order is **local** , then **radius** , and then **tacacs** . With the default authentication order, the authentication process occurs in the following sequence:

- The authentication process first checks whether a username and matching password are present in the running configuration on the local device.
- If local authentication fails, and if you have not configured authentication fallback (with the **auth-fallback** command), the authentication process stops. However, if you have configured authentication fallback, the authentication process next checks the RADIUS server. For this method to work, you must configure one or more RADIUS servers with the **system radius server** command. If a RADIUS server is reachable, the user is authenticated or denied access based on that server's RADIUS database. If a RADIUS server is unreachable and if you have configured multiple RADIUS servers, the authentication process checks each server sequentially, stopping when it is able to reach one of them. The user is then authenticated or denied access based on that server's RADIUS database.
- If the RADIUS server is unreachable (or all the servers are unreachable), the authentication process checks the TACACS+ server. For this method to work, you must configure one or more TACACS+ servers with the **system tacacs server** command. If a TACACS+ server is reachable, the user is authenticated or denied access based on that server's TACACS+ database. If a TACACS+ server is unreachable and if you have configured multiple TACACS+ servers, the authentication process checks each server sequentially, stopping when it is able to reach one of them. The user is then authenticated or denied access based on that server's TACACS+ database.
- If the TACACS+ server is unreachable (or all TACACS+ servers are unreachable), user access to the local Viptela device is denied.

To modify the default order, use the **auth-order** command:

```
Viptela(config-system-aaa)# auth-order (local | radius | tacacs)
```

Specify one, two, or three authentication methods in the preferred order, starting with the one to be tried first. If you configure only one authentication method, it must be **local** .

To have the "admin" user use the authentication order configured in the **auth-order** command, use the following command:

```
Viptela(config-system-aaa)# admin-auth-order
```

If you do not include this command, the "admin" user is always authenticated locally.

You can configure authentication to fall back to a secondary or tertiary authentication mechanism when the higher-priority authentication method fails to authenticate a user, either because the user has entered invalid credentials or because the authentication server is unreachable (or all the servers are unreachable):

```
Viptela(config-system-aaa)# auth-fallback
```

Fallback to a secondary or tertiary authentication mechanism happens when the higher-priority authentication server fails to authenticate a user, either because the credentials provided by the user are invalid or becasue the server is unreachable.

The following examples illustrate the default authentication behavior and the behavior when authentication fallback is enabled:

- If the authentication order is configured as **radius local** :
  - With the default authentication, local authentication is used only when all RADIUS servers are unreachable. If an authentication attempt via a RADIUS server fails, the user is not allowed to log in even if they have provided the correct credentials for local authentication.
  - With authentication fallback enabled, local authentication is used when all RADIUS servers are unreachable or when a RADIUS server denies access to a user.
- If the authentication order is configured as **local radius** :
  - With the default authentication, RADIUS authentication is tried when a username and matching password are not present in the running configuration on the local device.
  - With authentication fallback enabled, RADIUS authentication is tried when a username and matching password are not present in the running configuration on the local device. In this case, the behavior of two authentication methods is identical.

- If the authentication order is configured as **radius tacacs local** :
  - With the default authentication, TACACS+ is tried only when all RADIUS servers are unreachable, and local authentication is tried only when all TACACS+ servers are unreachable. If an authentication attempt via a RADIUS server fails, the user is not allowed to log in even if they have provided the correct credentials for the TACACS+ server. Similarly, if a TACACS+ server denies access, the user cannot log via local authentication.
  - With authentication fallback enabled, TACACS+ authentication is used when all RADIUS servers are unreachable or when a RADIUS server denies access a user. Local authentication is used next, when all TACACS+ servers are unreachable or when a TACACS+ server denies access to a user.

If a remote server validates authentication but does not specify a user group, the user is placed into the user group **basic** .

If a remote server validates authentication and specifies a user group (say, X), the user is placed into that user group only. However, if that user is also configured locally and belongs to a user group (say, Y), the user is placed into both the groups (X and Y).

If a remote server validates authentication and that user is not configured locally, the user is logged in to the vshell as the user **basic** , with a home directory of /home/basic.

If a remote server validates authentication and that user is configured locally, the user is logged in to the vshell under their local username (say, eve) with a home direction of /home/ *username* (so, /home/eve).

Configuring NAS Attributes

For RADIUS and TACACS+, you can configure Network Access Server (NAS) attributes for user authentication and authorization. To do this, you create a vendor-specific attributes (VSA) file, also called a RADIUS dictionary or a TACACS+ dictionary, on the RADIUS or TACACS+ server that contains the desired permit and deny commands for each user. The Viptela device retrieves this information from the RADIUS or TACACS+ server.

The VSA file must be named dictionary.viptela, and it must contain text in the following format:

```
localhost$ more dictionary.viptela
 # -*- text -*-
#
#  dictionary.viptela
#
#
# Version:      $Id$
#

VENDOR          Viptela                         41916

BEGIN-VENDOR    Viptela

ATTRIBUTE       Viptela-Group-Name              1    string
```

The Viptela software has three predefined user groups, as described above: **basic** , **netadmin** , and **operator** . These groups have the following permissions:

```
Viptela# show aaa usergroup
GROUP     USERS  TASK       PERMISSION
-----------------------------------------
basic     -      system     read
                 interface  read
netadmin  admin  system     read write
                 interface  read write
                 policy     read write
                 routing    read write
                 security   read write
operator  -      system     read
                 interface  read
```

```
                policy     read
                routing    read
                security   read
```

To create new user groups, use this command:

```
Viptela(config)# system aaa usergroup group-name task privilege
```

Here is a sample user configuration on a RADIUS server, which for FreeRADIUS would be in the file "users":

```
user1   Cleartext-password := "user123"
        Service-Type = NAS-Prompt-User,
        Viptela-Group-Name = operator,
```

Then in the dictionary on the RADIUS server, add a pointer to the VSA file:

```
$INCLUDE    /usr/share/freeradius/dictionary.viptela
```

For TACACS+, here is a sample configuration, which would be in the file tac_plus.conf:

```
group = test_group {
        default service = permit
        service = ppp protocol = ip {
              Viptela-Group-Name = operator
        }
}
user = user1 {
       pap = cleartext "user123"
       member = test_group
}
```

Additional Information

Configuring IEEE 802.1X and IEEE 802.11i Authentication
Role-Based Access with AAA
show aaa usergroup
show users

Configuring Time and Location

This article describes how to configure time and location services on Viptela devices.

Configure Network-Wide Time with NTP

To coordinate and synchronize time across all devices in the Viptela overlay network, configure the IP address or DNS server address of an NTP server on each device. The IP address must be an IPv4 address; it cannot be an IPv6 address. If necessary, specify the VPN through which the server is reachable.

```
Viptela(config)# system ntp server (dns-server-address | ipv4-address)
Viptela(config-system)# ntp server (dns-server-address | ipv4-address) vpn vpn-id
```

You can configure up to four NTP servers, and they must all be located or reachable in the same VPN. The software uses the server at the highest stratum level. If more than one server is at the same stratum level, you can configure the preference to use a specific server:

```
Viptela(config-ntp)# ntp server (dns-server-address | ipv4-address) prefer
```

You can configure an MD5 authentication key to use as a password to access an NTP server:

```
Viptela(config-system)# ntp keys
Viptela(config-keys)# authentication key-id md5 md5-key
```

key-id is a number that identifies the MD5 authentication key. It can be a number from 1 through 65535.

md5-key is the MD5 authentication key. You can enter it as cleartext or as an AES-encrypted key.

To use an MD5 authentication key for an NTP server, the key must be configured to be trusted:

```
Viptela(config-system)# ntp keys trusted key-id
```

Finally, associate the MD5 authentication key with the NTP time server:

```
Viptela(config-system)# ntp server (dns-server-address |ipv4-address) key key-id
```

You can configure NTP packets to exit from a specific interface on the router. The interface must be located in the same VPN as the NTP server. If it is not, the configuration is ignored.

```
Viptela(config-system)# ntp server (dns-server-address | ipv4-address) source-interface interface-name
```

The following example configures three NTP servers. One of the NTP servers is at the NTP pool project at the Network Time Foundation and uses no authentication. The other two are internal servers and are configured with MD5 authentication:

```
Viptela# show running-config system ntp
system
 ntp
  keys
   authentication 1001 md5 $4$KXLzYT9k6M8zj4BgLEFXKw==
   authentication 1002 md5 $4$KXLzYTxk6M8zj4BgLEFXKw==
   authentication 1003 md5 $4$KXLzYT1k6M8zj4BgLEFXKw==
   trusted 1001 1002
  !
  server 192.168.15.243
   key     1001
   vpn     512
   version 4
  exit
  server 192.168.15.242
   key     1002
   vpn     512
   version 4
  exit
  server us.pool.ntp.org
   vpn     512
   version 4
  exit
 !
!
```

Configuring NTP on a Viptela device allows that device to contact NTP servers to synchronize time. Other devices are allowed to ask a Viptela device for the time, but no devices are allowed to use the Viptela device as an NTP server.

Configure the Timezone

The default timezone on all Viptela devices is UTC.If your devices are located in multiple timezones (and even if they are not), we recommend that you use the default timezone, which is UTC, on all device so that the times in all logging and archive files are consistent.

To change the timezone on a device:

```
Viptela(config-system)# clock timezone timezone
```

Set the Time Locally

For Viptela devices that are part of a test or local network, you can set the time locally without using NTP because you do not need to ensure that time is synchronized across an entire network of devices. You can also set the time locally on any device as it is joining the

network, in addition to configuring an NTP server, and this time will be overwritten by the official NTP time once the device contacts the NTP server.

To set the local time and date, issue the following operational commands:

```
Viptela# clock set time hh:mm:ss[.sss]
Viptela# clock set date ccyy-mm-dd
```

You can also issue these commands as a single command:

```
Viptela# clock set date ccyy-mm-dd time hh:mm:ss[.sss]
```

or

```
Viptela# clock set time hh:mm:ss[.sss] date ccyy-mm-dd
```

To set the timezone, specify it in the configuration:

```
Viptela(config)# system clock timezone timezone
```

Configure a Device's Geographic Location

Configuring a device's geographic location by setting its latitude and longitude allows the device to be placed properly on the vManage NMS's network map.

To set a device's latitude and longitude:

```
Viptela(config-system)# gps-location latitude degrees.minutes-and-seconds longitude degrees.minutes-and-seconds
```

You can also set these values using two separate commands:

```
Viptela(config-system)# gps-location latitude degrees. minutes-and-seconds
Viptela(config-system)# gps-location longitude degrees. minutes-and-seconds
```

For example:

```
vEdge(config-system)# gps-location latitude 37.0000 longitude 122.0600

OR

vEdge(config-system)# gps-location latitude 37.000
vEdge(config-system)# gps-location longitude 122.0600

vEdge(config-system)# show full-configuration
system
 host-name         vEdge
 gps-location latitude 36.972
 gps-location longitude 122.0263
...
```

You can also configure a text description of the device's location:

```
Viptela(config-system)# location "description of location"
```

For example:

```
vEdge(config-system)# location "UCSC in Santa Cruz, California"
vEdge(config-system)# show full-configuration
system
 host-name         vEdge
 location          "UCSC in Santa Cruz, California"
 gps-location latitude 37.0000
 gps-location longitude 122.0600
...
```

Additional Information

show clock
show ntp associations
show ntp peer
show uptime

Configuring SNMP

This article describes how to enable SNMP on a Viptela device.

Enabling SNMP

By default, SNMP is disabled on Viptela devices. To enable it and provide support for SNMP Versions 1, 2, and 3:

```
Viptela(config)# snmp
Viptela(config-snmp)# no shutdown
```

Enabling SNMP allows the device to use MIBs, generate traps, and respond to requests from an SNMP walk application.

Configuring an SNMP View

To create an SNMP view, along with an OID, so that SNMP information is available to the SNMP server, configure an SNMP view and its corresponding OID subtree:

```
Viptela(config-snmp)# view string
Viptela(config-snmp)# oid oid-subtree
```

In the OID subtree, you can use the wildcard * (asterisk) in any position to match any value at that position.

The following example creates a view of the Internet portion of the SNMP MIB:

```
Viptela(config)# snmp view v2 oid 1.3.6.1
```

The following example creates a view of the private portion of the VIptela MIB:

```
Viptela(config)# snmp view viptela-private oid 1.3.6.1.4.1.41916
```

Configuring Access to an SNMP View

To require authentication privileges to access an SNMP view, configure SNMPv3. To do this, you configure authentication credentials for SNMPv3 users, and you configure groups of SNMP views and the authentication credentials required to access the views.

To configure authentication credentials for an SNMPv3 user, create a user and assign them an authentication level and a privacy level, depending on the authentication type you configure for the SNMP group (with the **snmp group** command, described below):

```
Viptela(config)# snmp user username
Viptela(config-user)# auth authentication
Viptela(config-user)# auth-password password
Viptela(config-user)# priv privacy
Viptela(config-user)# priv-password password
```

The username can be a string from 1 to 32 characters.

The authentication commands enable authentication privileges for the user. *authentication* can be either message digest 5 (**md5**) or SHA-2 message digest (**sha**). You can enter the password as a cleartext string or as an AES-encrypted key.

The privacy commands enable a privacy mechanism for the user. *privacy* can be either the Advanced Encryption Standard cipher algorithm used in cipher feedback mode, with a 128-bit key (**aes-cfb-128**). You can enter the password as a cleartext string or as an AES-encrypted key.

Then associate the SNMPv3 user with an SNMP group:

```
Viptela(config-user)# group group-name
```

group-name is the name of a group of views that you configure with the **snmp group** command.

To configure a group of views:

```
Viptela(config)# snmp group group-name authentication
Viptela(config-group)# view view-name
```

The group name can be a string from 1 to 32 characters.

The authentication to use to for the group can be one of the following:

- **auth-no-priv** —Authenticate using the HMAC-MD5 or HMAC-SHA algorithm. When you configure this authentication, users in this group must be configured with an authentication and an authentication password (with the **snmp user auth** and **auth-password** commands).
- **auth-priv** —Authenticate using the HMAC-MD5 or HMAC-SHA algorithm, and provide CBC DES 56-bit encryption. When you configure this authentication, users in this group must be configured with an authentication and an authentication password (with the **snmp user auth** and **auth-password** commands) and a privacy and privacy password (with the **snmp user priv** and **priv-password** commands).
- **no-auth-no-priv** —Authenticate based on a username. When you configure this authentication, you do not need to configure authentication or privacy credentials.

The view name is the name of a SNMP view that you configure with the **snmp view** command.

Here is an example configuration for SNMP users and groups:

```
vEdge(config-snmp)# show full-configuration
snmp
 no shutdown
 view v2
  oid 1.3.6.1
 !
 community private
  view          v2
  authorization read-only
 !
 group private-community auth-priv
  view v2
 !
 user noc-staff
  auth          md5
  auth-password $4$aCGzJjtS3/czj4BgLEFXKw==
  group         private-community
 !
!
```

Configuring Contact Parameters

For each Viptela device, you can configure its SNMP node name, physical location, and contact information for the person or entity responsible for the device:

```
Viptela(config)# snmp
Viptela(config-snmp)# name string
Viptela(config-snmp)# location string
Viptela(config-snmp)# contact string
```

If any of the strings include spaces, enclose the entire string in quotation marks (" ").

Configuring an SNMP Community

The SNMP community string defines the relationship between an SNMP server system and the client systems. This string acts like a password to control the clients' access to the server. To configure a community string, use the **community** command:

```
Viptela(config-snmp)# community name
Viptela(config-community-name)# authorization read-only
Viptela(config-community-name)# view string
```

The community name can be 1 through 32 characters long. It can include angle brackets (< and >). If the name includes spaces, enclose the entire name in quotation marks (" ").

Use the **view** command to specify the portion of the MIB tree to view. *string* is the name of a view record configured with the **snmp view** command, as described below.

The Viptela software supports the standard interfaces MIB, IF-MIB, and the system MIB (SNMPv2-MIB), which are automatically loaded onto the Viptela device when you install the Viptela software. For a list of enterprise MIBs, see the System and SNMP Overview . The MIBs supported by the Viptela software do not allow write operations, so you can configure only read-only authorization (which is the default authorization).

Configuring View Records

To configure a portion of an SNMP MIB to view, use the **view** command:

```
Viptela(config-snmp)# view string
Viptela(config-view)# oid oid-subtree [exclude]
```

For example, to view the Internet portion of the SNMP MIB configure the OID 1.3.6.1:

```
Viptela(config-snmp)# view v2 oid 1.3.6.1
```

To view the private portion of the Viptela MIB, configure the OID 1.3.6.1.4.1.41916.

Configuring SNMP Traps

SNMP traps are asynchronous notifications that a Viptela device sends to an SNMP management server. Traps notify the management server of events, whether normal or significant, that occur on the Viptela device. By default, SNMP traps are not sent to an SNMP server. Note that for SNMPv3, the PDU type for notifications ie either SNMPv2c inform (InformRequest-PDU) or trap (Trapv2-PDU).

To configure SNMP traps, you define the traps themselves and you configure the SNMP server that is to receive the traps.

To configure groups of traps to be collected on a Viptela device, use the **trap group** command:

```
Viptela(config-snmp)# trap group group-name
Viptela(config-group)# trap-type level severity
```

The *group-name* is a name of your choosing.

The *trap-type* can be one of those listed in the table below.

The severity level can be one or more of **critical** , **major** , and **minor** .

Severity Level

Trap Type

Critical

Major

Minor

all

All critical traps listed below.

All major traps listed below.

All minor traps listed below.

app-route

SLA_Change

bfd

BFD_State_Change

bridge

Bridge_Creation
Bridge_Deletion
Max_MAC_Reached

control

No_Active_vBond
No_Active_vSmart

Connection_Auth_Fail
Connection_State_Change
Connection_TLOC_IP_Change
vBond_State_Change

dhcp

Server_State_Change

Address_Assigned
Address_Released
Address_Renewed
Request_Rejected
Server_State_Change

hardware

EMMC_Fault
Fan_Fault
FanTray_Fault
Flash_Fault
PEM_Fault
PEM_State_Change
PIM_Fault
PIM_State_Change
SDCard_Fault
SFP_State_Change
TempSensor_Fault

TempSensor_State
USB_State_Change

omp

Data_Policy
Number_of_vSmarts_Change
Peer_State_Change
State_Change
TLOC_State_Change

policy

Access_List_Association_Status
Data_Policy_Association_Status
SLA_Violation_Pkt_Drop

SLA_Violation

routing

BGP_Peer_State_Change
OSPF_Interface_State_Change
OSPF_Neighbor_State_Change
PIM_Interface_State_Change
PIM_Neighbor_State_Change
PIM_Tunnel_State_Change

security

Clear_Installed_Certificate
Root_Cert_Chain_Uninstalled
vEdge_Entry_Added
vEdge_Entry_Removed
vEdge_Serial_File_Uploaded
vSmart_Entry_Added
vSmart_Entry_Removed
vSmart_Serial_File_Uploaded

Certificate_Installed
New_CSR_Generated
Root_Cert_Chain_Installed
Tunnel_IPSec_Manual_Rekey
Tunnel_IPSec_Rekey

system

AAA_Admin_Pwd_Change
Disk_Usage
Memory_Usage
Process_Restart
System_AAA_Login_Fail
System_Pseudo_Commit_Status
System_Reboot_Complete

Domain_ID_Change
Org_Name_Change
Reboot_Issued
Site_ID_Change
Software_Install_Status

System_Commit
System_IP_Change
System_Login_Change
System_Logout_Change

vpn

Interface_State_Change
VRRP_Group_State_Change

Route_Install_Fail
Tunnel_Install_Fail

wwan

Bearer_Change
Domain_State_Change
Reg_State_Change
SIM_State_Change

A single trap group can contain multiple trap types. In the configuration, specify one trap type per line, and each trap type can have one, two, or three severity levels. See the configuration example below for an illustration of the configuration process.

To configure the SNMP server to receive the traps, use the **trap target** command:

```
Viptela(config-snmp)# trap target vpn vpn-id ipv4-address udp-port
Viptela(config-target)# group-name name
Viptela(config-target)# community-name community-name
Viptela(config-target)# source-interface interface-name
```

For each SNMP server, specify the identifier of VPN where the server is located, the server's IPv4 address, and the UDP port on the server to connect to. When configuring the trap server's address, you must use an IPv4 address. You cannot use an IPv6 address.

In the **group-name** command, associate a previously configured trap group with the server. The traps in that group are sent to the SNMP server.

In the **community-name** command, associate a previously configure SNMP community with the SNMP server.

In the **source-interface** command, configure the interface to use to send traps to the SNMP server that is receiving the trap information. This interface cannot be a subinterface.

The following configuration example sends all traps to one SNMP server and only critical traps to another SNMP server. We configure two SNMP trap groups and the two target SNMP servers:

```
vEdge# config
Entering configuration mode terminal
vEdge(config)# snmp
vEdge(config-snmp)# view community-view
vEdge(config-view-community-view)# exit
vEdge(config-snmp)# community public
vEdge(config-community-public)# authorization read-only
vEdge(config-community-public)# view community-view
vEdge(config-community-public)# exit
vEdge(config-snmp)# trap group all-traps
vEdge(config-group-all-traps)# all level critical major minor
vEdge(config-group-all)# exit
vEdge(config-group-all-traps)# exit
vEdge(config-snmp)# trap group critical-traps
vEdge(config-group-critical-traps)# control level critical
vEdge(config-group-control)# exit
vEdge(config-group-critical-traps)# exit
vEdge(config-snmp)# trap target vpn 0 10.0.0.1 162
vEdge(config-target-0/10.0.0.1/162)# group-name all-traps
vEdge(config-target-0/10.0.0.1/162)# community-name public
```

```
vEdge(config-target-0/10.0.0.1/162)# exit
vEdge(config-snmp)# trap target vpn 0 10.0.0.2 162
vEdge(config-target-0/10.0.0.2/162)# group-name critical-traps
vEdge(config-target-0/10.0.0.2/162)# community-name public
vEdge(config-target-0/10.0.0.2/162)# exit
vEdge(config-snmp)# show full-configuration
snmp
 view community-view
 !
 community public
  view          community-view
  authorization read-only
 !
 trap target vpn 0 10.0.0.1 162
  group-name     all-traps
  community-name public
 !
 trap target vpn 0 10.0.0.2 162
  group-name     critical-traps
  community-name public
 !
 trap group all-traps
  all
   level critical major minor
  !
 !
 trap group critical-traps
  bfd
   level critical
  !
  control
   level critical
  !
  hardware
   level critical
  !
  omp
   level critical
  !
 !
!
vEdge(config-snmp)#
```

For each trap generated by a Viptela device, the device also generates a notification message. Use the show notification stream viptela command to display these messages. Here is an example of the command output. The first line of the output shows the time when the message was generated (the SNMP eventTime). The time is shown in UTC format. not in the device's local time. The second line of the notification contains a description of the event, and the third line indicates the severity level.

```
vEdge# show notification stream viptela
notification
 eventTime 2015-04-17T14:39:41.687272+00:00
 bfd-state-change
  severity-level major
  host-name vEdge
  system-ip 1.1.4.2
  src-ip 192.168.1.4
  dst-ip 108.200.52.250
  proto ipsec
  src-port 12346
  dst-port 12406
  local-system-ip 1.1.4.2
  local-color default
  remote-system-ip 1.1.9.1
  remote-color default
```

```
  new-state down
 !
!
notification
 eventTime 2015-04-17T15:12:20.435831+00:00
 tunnel-ipsec-rekey
  severity-level minor
  host-name vEdge
  system-ip 1.1.4.2
  color default
 !
!
notification
 eventTime 2015-04-17T16:56:50.314986+00:00
 system-login-change
  severity-level minor
  host-name vEdge
  system-ip 1.1.4.2
  user-name admin
  user-id 9890
 !
```

Additional Information

OID Repository
System, Interface, and SNMP CLI Reference
System and SNMP Overview
For Viptela enterprise MIBs, see the YANG Files for Netconf and Enterprise MIB Files section in the release notes for your software release.

Configuring Network Interfaces

In the Viptela overlay network design, interfaces are associated with VPNs. The interfaces that participate in a VPN are configured and enabled in that VPN. Each interface can be present only in a single VPN.

At a high level, for an interface to be operational, you must configure an IP address for the interface and mark it as operational (**no shutdown**). In practice, you always configure additional parameters for each interface.

You can configure up to 512 interfaces on a Viptela device. This number includes physical interfaces, loopback interfaces, and subinterfaces.

This article describes how to configure the general properties of WAN transport and service-side network interfaces. For information about how to configure specific interface types and properties—including cellular interfaces, DHCP, PPPoE, VRRP, and WLAN interfaces—see the links in the Additional Information section at the end of this article.

Configure Interfaces in the WAN Transport VPN (VPN 0)

VPN 0 is the WAN transport VPN. This VPN handles all control plane traffic, which is carried over OMP sessions, in the overlay network. For a Viptela device to participate in the overlay network, at least one interface must be configured in VPN 0, and at least one interface must connect to a WAN transport network, such as the Internet or an MPLS or a metro Ethernet network. This WAN transport interface is referred to as a tunnel interface. At a minimum, for this interface, you must configure an IP address, enable the interface, and set it to be a tunnel interface.

To configure a tunnel interface on a vSmart controller or a vManage NMS, you create an interface in VPN 0, assign an IP address or configure the interface to receive an IP address from DHCP, and mark it as a tunnel interface. The IP address can be either an IPv4 or IPv6 address. To enable dual stack, configure both address types. You can optionally associate a color with the tunnel.

Note: You can configure IPv6 addresses only on transport interfaces; that is, only in VPN 0.

```
vSmart/vManage(config)# vpn 0
vSmart/vManage(config-vpn-0)# interface interface-name
```

```
vSmart/vManage(config-interface)# [ip address prefix/length | ip dhcp-client [dhcp-distance number]
vSmart/vManage(config-interface)# [ipv6 address prefix/length | ipv6 dhcp-client [dhcp-distance number]
[dhcp-rapid-commit]
vSmart/vManage(config-interface)# no shutdown
vSmart/vManage(config-interface)# tunnel-interface
vSmart/vManage(config-tunnel-interface)# color color
vSmart/vManage(config-tunnel-interface)# [no] allow-service service
```

Tunnel interfaces on vEdge routers must have an IP address, a color, and an encapsulation type. The IP address can be either an IPv4 or IPv6 address. To enable dual stack, configure both address types.

```
vEdge(config)# vpn 0
vEdge(config-vpn-0)# interface interface-name
vEdge(config-interface)# [ip address prefix/length | ip dhcp-client [dhcp-distance number]
vEdge(config-interface)# [ipv6 address prefix/length | ipv6 dhcp-client [dhcp-distance number] [dhcp-
rapid-commit]
vEdge(config-interface)# no shutdown
vEdge(config-interface)# tunnel-interface
vEdge(config-tunnel-interface)# color color [restrict]
vEdge(config-tunnel-interface)# encapsulation (gre | ipsec)
vEdge(config-tunnel-interface)# [no] allow-service service
```

On vSmart controllers and vManage NMSs, *interface-name* can be either **eth** *number* or **loopback** *number* . Because vSmart controllers and vManage NMSs participate only in the overlay network's control plane, the only VPN that you can configure on these devices is VPN 0, and hence all interfaces are present only in this VPN.

On vEdge routers, *interface-name* can be **ge** *slot* / *port* , **gre** *number* , **ipsec** *number* , **loopback** *string* , **natpool** *number* , or **ppp** *number* .

To enable the interface, include the **no shutdown** command.

For the tunnel interface, you can configure a static IPv4 or IPv6 address, or you can configure the interface to receive its address from a DHCP server. To enable dual stack, configure both an IPv4 and an IPv6 address on the tunnel interface.

Color is a Viptela software construct that identifies the transport tunnel. It can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . The colors **metro-ethernet** , **mpls** , and **private1** through **private6** are referred to as *private colors* , because they use private addresses to connect to the remote side vEdge router in a private network. You can use these colors in a public network provided that there is no NAT device between the local and remote vEdge routers.

To limit the remote TLOCs that the local TLOC can establish BFD sessions with, mark the TLOC with the **restrict** option. When a TLOC is marked as restricted, a TLOC on the local router establishes tunnel connections with a remote TLOC only if the remote TLOC has the same color.

On a vSmart controller or vManage NMS, you can configure one tunnel interface. On a vEdge router, you can configure up to eight tunnel interfaces. This means that each vEdge router can have up to eight TLOCs.

On vEdge routers, you must configure the tunnel encapsulation. The encapsulation can be either IPsec or GRE. For IPsec encapsulation, the default MTU is 1442 bytes, and for GRE it is 1468 bytes, These values are a function of overhead required for BFD path MTU discovery, which is enabled by default on all TLOCs. (For more information, see Configuring Control Plane and Data Plane High Availability Parameters .) You can configure both IPsec and GRE encapsulation by including two **encapsulation** commands under the same **tunnel-interface** command. On the remote vEdge router, you must configure the same tunnel encapsulation type or types so that the two routers can exchange data traffic. Data transmitted out an IPsec tunnel can be received only by an IPsec tunnel, and data sent on a GRE tunnel can be received only by a GRE tunnel. The Viptela software automatically selects the correct tunnel on the destination vEdge router.

A tunnel interface allows only DTLS, TLS, and, for vEdge routers, IPsec traffic to pass through the tunnel. To allow additional traffic to pass without having to create explicit policies or access lists, enable them by including one **allow-service** command for each service. You can also explicitly disallow services by including the **no allow-service** command. Note that services affect only physical interfaces. You can allow or disallow these services on a tunnel interface:

Service	vEdge Router	vManage NMS	vSmart Controller
all (Overrides any commands that allow or disallow individual services)	X	X	X

bgp	X	—	—
dhcp (for DHCPv4 and DHCPv6)	X	—	—
dns	X	—	—
https	X	X	—
icmp	X	X	X
netconf	—	X	—
ntp	X	—	—
ospf	X	—	—
sshd	X	X	X
stun	X	X	X

The **allow-service stun** command pertains to allowing or disallowing a Viptela device to generate requests to a generic STUN server so that the device can determine whether it is behind a NAT and, if so, what kind of NAT it is and what the device's public IP address and public port number are. On a vEdge router that is behind a NAT, you can also have tunnel interface to discover its public IP address and port number from the vBond controller:

```
vEdge(config-tunnel-interface)# vbond-as-stun-server
```

With this configuration, the vEdge router uses the vBond orchestrator as a STUN server, so the router can determine its public IP address and public port number. (With this configuration, the router cannot learn the type of NAT that it is behind.) No overlay network control traffic is sent and no keys are exchanged over tunnel interface configured to the the vBond orchestrator as a STUN server. However, BFD does come up on the tunnel, and data traffic can be sent on it. Because no control traffic is sent over a tunnel interface that is configured to use the vBond orchestrator as a STUN server, you must configure at least one other tunnel interface on the vEdge router so that it can exchange control traffic with the vSmart controller and the vManage NMS.

On a vEdge router, services that you configure on a tunnel interface act as implicit access lists (ACLs). If you explicitly configure ACLs on a tunnel interface, with the **policy access-list** command, the handling of packets matching both implicit and explict ACLs depends on the exact configuration. For more information, see Configuring Localized Data Policy for IPv4 or Configuring Localized Data Policy for IPv6 .

For each transport tunnel on a vEdge router and for each encapsulation type on a single transport tunnel, the Viptela software creates a TLOC, which consists of the router' system IP address, the color, and the encapsulation. The OMP session running on the tunnel sends the TLOC, as a TLOC route, to the vSmart controller, which uses it to determine the overlay network topology and to determine the best paths for data traffic across the overlay network.

To display information about interfaces in the WAN transport VPN that are configured with IPv4 addresses, use the **show interface** command. For example:

```
vEdge# show interface vpn 0
                                  IF      IF
TCP
                                  ADMIN   OPER    ENCAP                                    SPEED
MSS                  RX       TX
VPN  INTERFACE  IP ADDRESS        STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR            MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
----------------------------------------
0    ge0/1      10.0.5.21/24      Up      Up      null   transport  1500  00:0c:29:6c:30:c1  10     full
0       0:04:03:41  260025   260145
0    ge0/2      -                 Down    Up      null   service    1500  00:0c:29:6c:30:cb  10     full
0       0:04:03:41  3506     1
0    ge0/3      -                 Down    Up      null   service    1500  00:0c:29:6c:30:d5  10     full
0       0:04:03:41  260      1
0    ge0/4      -                 Down    Up      null   service    1500  00:0c:29:6c:30:df  10     full
0       0:04:03:41  260      1
```

```
0    ge0/5      -                 Down    Up      null   service    1500  00:0c:29:6c:30:e9  10     full
0       0:04:03:41  260      1
0    ge0/6      10.0.7.21/24      Up      Up      null   service    1500  00:0c:29:6c:30:f3  10     full
0       0:04:03:41  265      2
0    ge0/7      10.0.100.21/24    Up      Up      null   service    1500  00:0c:29:6c:30:fd  10     full
0       0:04:03:41  278      2
0    system     172.16.255.21/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:04:03:37  0        0
```

To display information for interfaces configured with IPv6 addresses, use the **show ipv6 interface** command. For example:

```
vEdge# show ipv6 interface vpn 0

                                        IF      IF
TCP
                AF                      ADMIN   OPER    ENCAP                                   SPEED
MSS                 RX       TX
VPN  INTERFACE  TYPE  IPV6 ADDRESS      STATUS  STATUS  TYPE   PORT TYPE  MTU  HWADDR             MBPS
DUPLEX  ADJUST  UPTIME      PACKETS   PACKETS  LINK LOCAL ADDRESS
-------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------
0    ge0/1      ipv6  2001::a00:1a0b/120 Up      Up      null   service    1500 00:0c:29:ab:b7:62  1000
full    1420    0:01:30:00  2        6        fe80::20c:29ff:feab:b762/64
0    ge0/2      ipv6  2001::a00:50b/120  Up      Up      null   service    1500 00:0c:29:ab:b7:6c  1000
full    1420    0:01:30:00  21       5        fe80::20c:29ff:feab:b76c/64
0    ge0/3      ipv6  fd00:1234::/16     Up      Up      null   service    1500 00:0c:29:ab:b7:76  1000
full    1420    0:01:08:33  0        8        fe80::20c:29ff:feab:b776/64
0    ge0/4      ipv6  -                  Up      Up      null   service    1500 00:0c:29:ab:b7:80  1000
full    1420    0:01:30:00  18       5        fe80::20c:29ff:feab:b780/64
0    ge0/5      ipv6  -                  Down    Up      null   service    1500 00:0c:29:ab:b7:8a  1000
full    1420    0:01:44:19  1        1        fe80::20c:29ff:feab:b78a/64
0    ge0/6      ipv6  -                  Down    Up      null   service    1500 00:0c:29:ab:b7:94  1000
full    1420    0:01:44:19  0        1        fe80::20c:29ff:feab:b794/64
0    ge0/7      ipv6  -                  Up      Up      null   service    1500 00:0c:29:ab:b7:9e  1000
full    1420    0:01:43:02  55       5        fe80::20c:29ff:feab:b79e/64
0    system     ipv6  -                  Up      Up      null   loopback   1500 00:00:00:00:00:00  10
full    1420    0:01:29:31  0        0        -
0    loopback1  ipv6  2001::a00:6501/128 Up      Up      null   transport  1500 00:00:00:00:00:00  10
full    1420    0:03:49:09  0        0        -
0    loopback2  ipv6  2001::a00:6502/128 Up      Up      null   transport  1500 00:00:00:00:00:00  10
full    1420    0:03:49:05  0        0        -
0    loopback3  ipv6  2001::a00:6503/128 Up      Up      null   transport  1500 00:00:00:00:00:00  10
full    1420    0:03:49:01  0        0        -
0    loopback4  ipv6  2001::a00:6504/128 Up      Up      null   transport  1500 00:00:00:00:00:00  10
full    1420    0:03:48:54  0        0        -
```

In the command output, a port type of "transport" indicates that the interface is configured as a tunnel interface, and a port type of "service" indicates that the interface is not configured as a tunnel interface and can be used for data plane traffic. The port type for the system IP address interface is "loopback".

Associate a Carrier Name with a Tunnel Interface

To associate a carrier name or private network identifier with a tunnel interface, use the **carrier** command. *carrier-name* can be **default** and **carrier1** through **carrier8:**

```
Viptela(config)# vpn 0
Viptela(config-vpn-0)# interface interface-name
Viptela(config-interface)# tunnel-interface
Viptela(config-tunnel-interface)# carrier carrier-name
```

Limit Keepalive Traffic on a Tunnel Interface

By default, Viptela devices send a Hello packet once per second to determine whether the tunnel interface between two devices is still operational and to keep the tunnel alive. The combination of a hello interval and a hello tolerance determines how long to wait before declaring a DTLS or TLS tunnel to be down. The default hello interval is 1 second, and the default tolerance is 12 seconds. With these default values, if no Hello packet is received within 11 seconds, the tunnel is declared down at 12 seconds.

If the hello interval or the hello tolerance, or both, are different at the two ends of a DTLS or TLS tunnel, the tunnel chooses the interval and tolerance as follows:

- For a tunnel connection between two controller devices, the tunnel uses the lower hello interval and the higher tolerance interval for the connection between the two devices. (Controller devices are vBond controllers, vManage NMSs, and vSmart controllers.) This choice is made in case one of the controllers has a slower WAN connection. The hello interval and tolerance times are chosen separately for each pair of controller devices.
- For a tunnel connection between a vEdge router and any controller device, the tunnel uses the hello interval and tolerance times configured on the router. This choice is made to minimize the amount traffic sent over the tunnel, to allow for situations where the cost of a link is a function of the amount of traffic traversing the link. The hello interval and tolerance times are chosen separately for each tunnel between a vEdge router and a controller device.

To minimize the amount of keepalive traffic on a tunnel interface, increase the Hello packet interval and tolerance on the tunnel interface:

```
vEdge(config-tunnel-interface)# hello-interval milliseconds
vEdge(config-tunnel-interface)# hello-tolerance seconds
```

The default hello interval is 1000 milliseconds, and it can be a time in the range 100 through 600000 milliseconds (10 minutes). The default hello tolerance is 12 seconds, and it can be a time in the range 12 through 600 seconds (10 minutes). The hello tolerance interval must be at most one-half the OMP hold time. The default OMP hold time is 60 seconds, and you configure it with the **omp timers holdtime** command.

Limit the HTTPS Connections to a vManage Server

By default, a vManage application server accepts a maximum of 50 HTTPS connections from users in the overlay network. To modify the maximum number of HTTPS connections that can be established:

```
vManage(config)# vpn 0 interface interface-name tunnel-interface control-connections number
```

The number can be from 1 through 512.

Configure Multiple Tunnel Interfaces on a vEdge Router

On a vEdge router, you can configure up to eight tunnel interfaces in the transport interface (VPN 0). This means that each vEdge router can have up to eight TLOCs.

When a vEdge router has multiple TLOCs, each TLOC is preferred equally and traffic to each TLOC is weighted equally, resulting in ECMP routing. ECMP routing is performed regardless of the encapsulation used on the transport tunnel, so if, for example, a router has one IPsec and one GRE tunnel, with ECMP traffic is forwarded equally between the two tunnels. You can change the traffic distribution by modifying the preference or the weight, or both, associated with a TLOC. (Note that you can also affect or change the traffic distribution by applying a policy on the interface that affects traffic flow.)

```
vEdge(config)# vpn 0
vEdge(config-vpn-0)# interface interface-name
vEdge(config-tunnel-interface) encapsulation (gre | ipsec)
vEdge(config-encapsulation)# preference number
vEdge(config-encapsulation)# weight number
```

The **preference** command controls the preference for directing traffic to a tunnel. The preference can be a value from 0 through 4294967295 ($2^{32} - 1$), and the default value is 0. A higher value is preferred over a lower value.

When a vEdge router has two or more tunnels, if all the TLOCs have the same preference and no policy is applied that affects traffic flow, all the TLOCs are advertised into OMP. When the router transmits or receives traffic, it distributes traffic flows evenly among the tunnels, using ECMP.

When a vEdge router has two or more tunnels, if the TLOCs all have different preferences and no policy is applied that affects traffic flow, only the TLOC with the highest preference is advertised into OMP. When the router transmits or receives traffic, it sends the traffic only to the TLOC with the highest preference. When there are three or more tunnels and two of them have the same preference, traffic flows are distributed evenly between these two tunnels.

A remote vEdge router trying to reach one of these prefixes selects which TLOC to use from the set of TLOCs that have been advertised. So, for example, if a remote router selects a GRE TLOC on the local router, the remote router must have its own GRE TLOC to be able to reach the prefix. If the remote router has no GRE TLOC, it is unable to reach the prefix. If the remote router has a single GRE TLOC, it selects that tunnel even if there is an IPsec TLOC with a higher preference. If the remote router has multiple GRE TLOCs, it selects from among them, choosing the one with the highest preference or using ECMP among GRE TLOCs with equal preference, regardless of whether there is an IPsec TLOC with a higher preference.

The **weight** command controls how traffic is balanced across multiple TLOCs that have equal preferences values. The weight can be a value from 1 through 255, and the default is 1. When the weight value is higher, the router sends more traffic to the TLOC. You typically set the weight based on the bandwidth of the TLOC. When a router has two or more TLOCs, all with the highest equal preference value, traffic distribution is weighted according to the configured weight value. For example, if TLOC A has weight 10, and TLOC B has weight 1, and both TLOCs have the same preference value, then roughly 10 flows are sent out TLOC A for every 1 flow sent out TLOC B.

Configure Control Plane High Availability

A highly available Viptela network contains two or more vSmart controllers in each domain. A Viptela domain can have up to eight vSmart controllers, and each vEdge router, by default, connects to two of them. You change this value on a per-tunnel basis:

```
vEdge(config-tunnel-interface)# max-controllers number
```

When the number of vSmart controllers in a domain is greater than the maximum number of controllers that a domain's vEdge routers are allowed to connect to, the Viptela software load-balances the connections among the available vSmart controllers.

Tip : To maximize the efficiency of the load-balancing among vSmart controllers, use sequential numbers when assigning system IP addresses to the vEdge routers in the domain. One example of a sequential numbering schemes is 172.1.1.1, 172.1.1.2, 172.1.1.3, and so forth. Another is 172.1.1.1, 172.1.2.1, 172.1.3.1, and so forth.

Configure Other WAN Interface Properties

You can modify the distribution of data traffic across transport tunnels by applying a data policy in which the action sets TLOC attributes (IP address, color, and encapsulation) to apply to matching data packets. For more information, see Configuring Centralized Data Policy .

Configure the System Interface

For each Viptela device, you configure a system interface with the **system system-ip** command. The system interface's IP address is a persistent address that identifies the Viptela device. It is similar to a router ID on a regular router, which is the address used to identify the router from which packets originated.

```
Viptela(config)# system system-ip ipv4-address
```

Specify the system IP address as an IPv4 address in decimal four-part dotted notation. Specify just the address; the prefix length (/32) is implicit.

The system IP address can be any IPv4 address except for 0.0.0.0/8, 127.0.0.0/8, and 224.0.0.0/4, and 240.0.0.0/4 and later. Each device in the overlay network must have a unique system IP address. You cannot use this same address for another interface in VPN 0.

The system interface is placed in VPN 0, as a loopback interface named **system** . Note that this is not the same as a loopback address that you configure for an interface.

To display information about the system interface, use the **show interface** command. For example:

```
vEdge# show running-config system system-ip
system
 system-ip 172.16.255.11
!
vEdge# show interface vpn 0
                                    IF      IF
TCP
                                    ADMIN   OPER    ENCAP                                       SPEED
MSS                  RX       TX
VPN   INTERFACE  IP ADDRESS        STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR              MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------
0     ge0/1      10.0.26.11/24     Up      Up      null   service    1500  00:0c:29:ab:b7:62  1000   full
1420    0:10:32:16  1606     8
0     ge0/2      10.0.5.11/24      Up      Up      null   transport  1500  00:0c:29:ab:b7:6c  1000   full
1420    0:10:32:16  307113   303457
0     ge0/3      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:76  1000   full
1420    0:10:47:49  1608     0
0     ge0/4      10.0.7.11/24      Up      Up      null   service    1500  00:0c:29:ab:b7:80  1000   full
1420    0:10:32:16  1612     8
0     ge0/5      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:8a  1000   full
1420    0:10:47:49  1621     0
0     ge0/6      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:94  1000   full
1420    0:10:47:49  1600     0
0     ge0/7      10.0.100.11/24    Up      Up      null   service    1500  00:0c:29:ab:b7:9e  1000   full
1420    0:10:47:31  3128     1165
0     system     172.16.255.11/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
1420    0:10:31:58  0        0
```

The system IP address is used as one of the attributes of the OMP TLOC. Each TLOC is uniquely identified by a 3-tuple comprising the system IP address, a color, and an encapsulation. To display TLOC information, use the **show omp tlocs** command.

For device management purposes, it is recommended as a best practice that you also configure the same system IP address on a loopback interface that is located in a service-side VPN that is an appropriate VPN for management purposes. You use a loopback interface because it is always reachable when the router is operational and when the overlay network is up. If you were to configure the system IP address on a physical interface, both the router and the interface would have to be up for the router to be reachable. You use a service-side VPN because it is reachable from the data center. Service-side VPNs are VPNs other than VPN 0 (the WAN transport VPN) and VPN 512 (the management VPN), and they are used to route data traffic.

Here is an example of configuring the system IP address on a loopback interface in VPN 1:

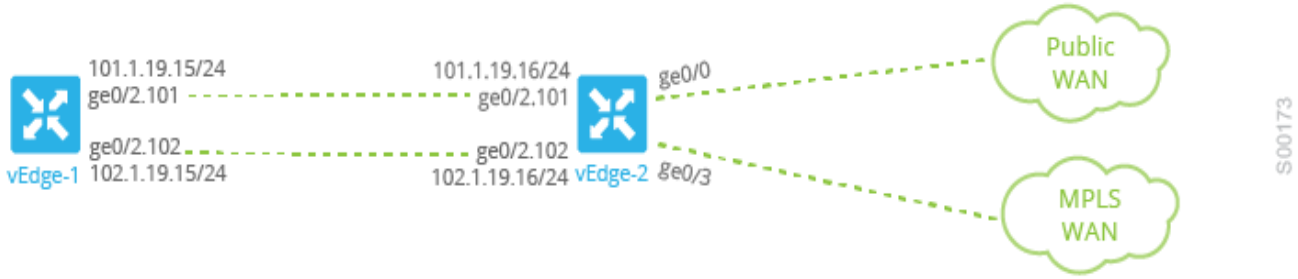
```
vEdge# config
Entering configuration mode terminal
vEdge(config)# vpn 1
vEdge(config-vpn-1)# interface loopback0 ip address 172.16.255.11/32
vEdge(config-vpn-1)# no shutdown
vEdge(config-interface-loopback0)# commit and-quit
Commit complete.
vEdge# show interface
                                    IF      IF
TCP
                                    ADMIN   OPER    ENCAP                                       SPEED
MSS                  RX       TX
VPN   INTERFACE  IP ADDRESS        STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR              MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------
0     ge0/1      10.0.26.11/24     Up      Up      null   service    1500  00:0c:29:ab:b7:62  1000   full
```

```
1420    0:10:27:33  1597     8
0    ge0/2      10.0.5.11/24      Up      Up      null   transport  1500  00:0c:29:ab:b7:6c  1000   full
1420    0:10:27:33  304819   301173
0    ge0/3      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:76  1000   full
1420    0:10:43:07  1599     0
0    ge0/4      10.0.7.11/24      Up      Up      null   service    1500  00:0c:29:ab:b7:80  1000   full
1420    0:10:27:33  1603     8
0    ge0/5      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:8a  1000   full
1420    0:10:43:07  1612     0
0    ge0/6      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:94  1000   full
1420    0:10:43:07  1591     0
0    ge0/7      10.0.100.11/24    Up      Up      null   service    1500  00:0c:29:ab:b7:9e  1000   full
1420    0:10:42:48  3118     1164
0    system     172.16.255.11/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
1420    0:10:27:15  0        0
1    ge0/0      10.2.2.11/24      Up      Up      null   service    1500  00:0c:29:ab:b7:58  1000   full
1420    0:10:27:30  5734     4204
1    loopback0  172.16.255.11/32  Up      Up      null   service    1500  00:00:00:00:00:00  10     full
1420    0:00:00:28  0        0
512  eth0       10.0.1.11/24      Up      Up      null   service    1500  00:50:56:00:01:0b  1000   full
0       0:10:43:03  20801    14368
```

Extend the WAN Transport VPN

When two vEdge routers are collocated at a physical site that has only one WAN circuit, you can configure the vEdge router that is not connected to the circuit to be able to establish WAN transport tunnels through the other router's TLOCs. In this way, you extend the WAN transport VPN so that both routers can establish tunnel interfaces, and hence can establish independent TLOCs, in the overlay network. (Note that you can configure the two routers themselves with different site identifiers.)

The following figure illustrates a site with two vEdge routers. vEdge-2 has two WAN circuits, one to the Internet and a second to a private MPLS network, and so has two TLOCs. By itself, vEdge1 has no TLOCs. You can configure vEdge-2 to extend its WAN transport VPN to vEdge1 so that vEdge-1 can participate independently in the overlay network.



When you extend the WAN transport VPN, no BFD sessions are established between the two collocated vEdge routers.

You cannot configure TLOC extensions on cellular (LTE) interfaces.

To extend the WAN transport VPN, you configure the interface between the two routers:

- For the router that is not connected to the circuit, you configure a standard tunnel interface in VPN 0.
- For the router that is physically connected to the WAN or private transport, you associate the physical interface that connects to the circuit, configuring this in VPN 0 but not in a tunnel interface.

To configure the non-connected router (vEdge-1 in the figure above), create a tunnel interface in VPN 0 on the physical interface to the connected router.

```
vEdge-1(config-vpn-0)# interface geslot/port
vEdge-1(config-interface)# ip address prefix/length
```

```
vEdge-1(config-interface)# no shutdown
vEdge-1(config-interface)# mtu number
vEdge-1(config-interface)# tunnel-interface
vEdge-1(config-tunnel-interface)# color color
```

For the router connected to the WAN or private transport (vEdge-2 in the figure above), configure the interface that connects to the non-connected router, again in VPN 0:

```
vEdge-2(config-vpn-0)# interface geslot/port
vEdge-2(config-interface)# ip address prefix/length
vEdge-2(config-interface)# tloc-extension geslot/port
vEdge-2(config-interface)# no shutdown
vEdge-2(config-interface)# mtu number
```

The physical interface in the **interface** command is the one that connects to the other router.

The **tloc-extension** command creates the binding between the non-connected router and the WAN or private network. In this command, you specify the physical interface that connects to the WAN or private network circuit.

If the circuit connects to a public network:

- Configure a NAT on the public-network-facing interface on the vEdge router. The NAT configuration is required because the two vEdge routers are sharing the same transport tunnel.
- Configure a static route on the non-connected router to the TLOC-extended interface on the router connected to the public network.

If the circuit connects to a private network, such as an MPLS network:

- Enable routing on the non-connected router so that the interface on the non-connected router is advertised into the private network.
- Depending on the routing protocol you are using, enable either OSPF or BGP service on the non-connected router interface so that routing between the non-connected and the connected routers comes up. To do this, use the **allow-service** command.

You cannot extend a TLOC configured on a loopback interface, that is, when you use a loopback interface to connect to the public or private network. You can extend a TLOConly on a physical interface.

If one of the routers is connected to two WAN transports (such as the Internet and an MPLS network), create subinterfaces between the two routers, creating the tunnel on the subinterface. The subinterfaces on the two routers must be in the same subnet. Because you are using a subinterface, the interface's MTU must be at least 4 bytes less than the physical MTU.

By default, routers at one site form BFD tunnels only with routers at remote sites. If you want the routers at the same site to form BFD tunnels between them, enable the formation of these tunnels:

```
vEdge(config)# system allow-same-site-tunnels
```

Here is a sample configuration that corresponds to the figure shown above. Because the router vEdge-2 connects to two transports, we create subinterfaces between the vEdge-1 and vEdge-2 routers. One subinterface binds to the Internet circuit, and the second one binds to the MPLS connection.

```
vEdge-1# show running-config vpn 0
interface ge0/2.101
  ip address 101.1.19.15/24
  mtu 1496
  tunnel-interface
    color lte
  ...
  !
  no shutdown
!
interface ge0/2.102
  ip address 102.1.19.15/24
  mtu 1496
  tunnel-interface
    color mpls
  ...
```

```
  !
  no shutdown
!
ip route 0.0.0.0/0 101.1.19.16
vEdge-2# show running-config vpn 0
interface ge0/0
  ip address 172.16.255.2
  tunnel-interface
    color lte
  ...
  !
  no shutdown
!
interface ge0/3
  ip address 172.16.255.16
  tunnel-interface
    color mpls
  ...
  !
  no shutdown
!
interface ge0/2.101
  ip address 101.1.19.16/24
  mtu 1496
  tloc-extension ge0/0
  no shutdown
!
interface ge0/2.102
  ip address 102.1.19.16/24
  mtu 1496
  tloc-extension ge0/3
  no shutdown
!
```

For this example configuration, vEdge-1 establishes two control connections to each vSmart controller in the overlay network—one connection for the LTE tunnel and the second for the MPLS tunnel. These control connections are separate and independent from those established on vEdge-2. The following output shows the control connections on vEdge-1 in a network with two vSmart controllers:

```
vEdge-1# show control connections
                                                                                    PEER
PEER                                                CONTROLLER
PEER     PEER     PEER             SITE        DOMAIN      PEER             PRIVATE  PEER
PUBLIC                                              GROUP
TYPE     PROTOCOL SYSTEM IP        ID          ID          PRIVATE IP       PORT     PUBLIC IP
PORT    LOCAL COLOR      STATE      UPTIME           NAME
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------
vsmart   dtls     172.16.255.19    100         1           10.0.5.19        12346    10.0.5.19
12346   lte              up              0:00:18:43 default
vsmart   dtls     172.16.255.19    100         1           10.0.5.19        12346    10.0.5.19
12346   mpls             up              0:00:18:32 default
vsmart   dtls     172.16.255.20    200         1           10.0.12.20       12346    10.0.12.20
12346   lte              up              0:00:18:38 default
vsmart   dtls     172.16.255.20    200         1           10.0.12.20       12346    10.0.12.20
12346   mpls             up              0:00:18:27 default
```

You can verify that the two vEdge routers have established no BFD sessions between them. On vEdge-1, we see no BFD sessions to vEdge-2 (system IP address 172.16.255.16):

```
vEdge-1# show bfd sessions
                                  SOURCE TLOC   REMOTE TLOC                       DST PUBLIC       DST
PUBLIC          DETECT      TX                        TRANSI-
SYSTEM IP        SITE ID  STATE   COLOR         COLOR            SOURCE IP        IP               PORT
ENCAP  MULTIPLIER  INTERVAL(msec)  UPTIME     TIONS
```

```
-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
172.16.255.11    100      up      lte           lte              101.1.19.15      10.0.101.1       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      lte           3g               101.1.19.15      10.0.101.2       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      lte           gold             101.1.19.15      10.0.101.3       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      lte           red              101.1.19.15      10.0.101.4       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      mpls          lte              102.1.19.15      10.0.101.1       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      mpls          3g               102.1.19.15      10.0.101.2       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      mpls          gold             102.1.19.15      10.0.101.3       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.11    100      up      mpls          red              102.1.19.15      10.0.101.4       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.14    400      up      lte           lte              101.1.19.15      10.1.14.14       12360
ipsec  20          1000           0:00:20:26   0
172.16.255.14    400      up      mpls          lte              102.1.19.15      10.1.14.14       12360
ipsec  20          1000           0:00:20:26   0
172.16.255.21    100      up      lte           lte              101.1.19.15      10.0.111.1       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.21    100      up      lte           3g               101.1.19.15      10.0.111.2       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.21    100      up      mpls          lte              102.1.19.15      10.0.111.1       12346
ipsec  20          1000           0:00:20:26   0
172.16.255.21    100      up      mpls          3g               102.1.19.15      10.0.111.2       12346
ipsec  20          1000           0:00:20:26   0
```

Configure Interfaces in the Management VPN (VPN 512)

On all Viptela devices, VPN 512 is used, by default, for out-of-band management, and its configuration is part of the factory-default configuration. The interface type for management interfaces is **mgmt** , and the initial address for the interface is 192.168.1.1.

```
Viptela# show running-config vpn 512
vpn 512
 interface mgmt0
  ip dhcp-client
  no shutdown
 !
!
```

To display information about the configured management interfaces, use the **show interface** command. For example:

```
vEdge# show interface vpn 512

                              IF      IF
TCP
                              ADMIN   OPER    ENCAP  PORT                              SPEED
MSS                 RX       TX
VPN  INTERFACE  IP ADDRESS     STATUS  STATUS  TYPE   TYPE     MTU   HWADDR             MBPS   DUPLEX
ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------
512  mgmt0      192.168.1.1/24  Up      Up      null   service  1500  00:50:56:00:01:1f  1000   full    0
0:04:08:01  1131     608
```

Note that VPN 512 is not a routable VPN. If you need a routable management VPN, create a VPN with a number other than 512.

Configure Service-Side Interfaces for Carrying Data Traffic

On vEdge routers, the VPNs other than 0 and 512 are service-side VPNs, and the interfaces in these VPNs connect the router to service-side LANs and WLANs. These interfaces are the interfaces that carry data traffic between vEdge routers and sites across the overlay network. At a minimum, for these interfaces, you must configure an IPv4 address, and you must enable the interface:

```
vEdge(config)# vpn vpn-id
vEdge(config-vpn)# interface geslot/port
vEdge(config-interface)# ip address prefix/length
vEdge(config-interface)# no shutdown
```

For service-side interfaces, you can configure up to four secondary IP addresses:

```
vEdge(config)# vpn vpn-id
vEdge(config-vpn)# interface geslot/port
vEdge(config-interface)# ip secondary-address ipv4-address
```

To display information about the configured data traffic interfaces, use the **show interface** command. For example:

```
vEdge# show interface vpn 1

                              IF      IF
TCP
                              ADMIN   OPER    ENCAP  PORT                              SPEED
MSS                 RX       TX
VPN  INTERFACE  IP ADDRESS     STATUS  STATUS  TYPE   TYPE     MTU   HWADDR             MBPS   DUPLEX
ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
-----------------------------------
1    ge0/1      10.192.1.1/28  Up      Up      null   service  1500  00:0c:bd:05:f0:84  100    full    0
1:05:44:07  399      331
1    loopback1  1.1.1.1/32     Up      Up      null   service  1500  00:00:00:00:00:00  10     full    0
1:05:44:07  0        0
```

For some protocols, you specify an interface as part of the protocol's configuration. In these cases, the interface used by the protocol must be the same as one of the interfaces configured in the VPN. As example is OSPF, where you place interfaces in OSPF areas. In this example, the interface **ge0/0** is configured in VPN 1, and this interface is configured to be in the OSPF backbone area:

```
vEdge# show running-config vpn 1
vpn 1
 router
  ospf
   router-id 172.16.255.21
   timers spf 200 1000 10000
   redistribute static
   redistribute omp
   area 0
    interface ge0/0
    exit
   exit
  !
 !
 interface ge0/0
  ip address 10.2.3.21/24
  no shutdown
 !
!
```

Configure Subinterfaces and VLANs

You can configure IEEE 802.1Q VLANs on vEdge routers. In such VLANs, physical interfaces are divided into subinterfaces. When you configure a subinterface, the interface name has the format **ge** *slot* / *port* . *vlan-number.* The VLAN number, *vlan-number* , can be in the range 1 through 4094.

As with all interfaces, the subinterface must be activated, by configuring it with the **no shutdown** command.

To accommodate the 32-bit field added to packets by the 802.1Q protocol, you must also configure the MTU for VLAN subinterfaces to be at least 4 bytes smaller than the MTU of the physical interface. You do this using the **mtu** command. The default MTU on a physical interface is 1500 bytes by default, so the subinterface's MTU here can be no larger than 1496 bytes.

For subinterfaces to work, you must configure the physical interface in VPN 0 and activate it with a **no shutdown** command. If the physical interface goes down for any reason, all its subinterfaces also go down. If you shut down the subinterface with a **shutdown** command, the operational status remains up as long as the physical interface is up.

You can place the VLANs associated with a single physical interface into multiple VPNs. Each individual subinterface can be present only in a single VPN.

Here is an example of a minimal VLAN configuration. The VLANs are configured on subinterfaces **ge0/6.2** and **ge0/6.3** in VPN 1, and they are associated with the physical interface **ge0/6** in VPN 0.

```
vEdge# show running-config vpn 1
vpn 1
 interface ge0/6.2
  mtu      1496
  no shutdown
 !
interface ge0/6.3
  mtu      1496
  no shutdown
 !
!
vEdge# show running-config vpn 0
vpn 0
 interface ge0/0
  ip dhcp-client
  tunnel-interface
   encapsulation ipsec
   no allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-serivce ospf
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
 interface ge0/6
  ip address 57.0.1.15/24
  no shutdown
 !
!
```

The output of the **show interface** command shows the physical interface and the subinterfaces. The Encap Type column shows that the subinterfaces are VLAN interfaces, and the MTU column shows that the physical interface has an MTU size of 1500 bytes, while the MTU of the subinterfaces is 1496 bytes, so 4 bytes less.

```
vEdge# show interface

                                  IF      IF
TCP
                                  ADMIN   OPER    ENCAP                                    SPEED
MSS                 RX       TX
VPN  INTERFACE  IP ADDRESS        STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR            MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
```

```
----------------------------------------
0    ge0/0      10.1.15.15/24     Up      Up      null   transport  1500  00:0c:29:7d:1e:fe  10     full
0       0:04:32:28  289584   289589
0    ge0/1      10.1.17.15/24     Up      Up      null   service    1500  00:0c:29:7d:1e:08  10     full
0       0:04:32:28  290      2
0    ge0/2      -                 Down    Up      null   service    1500  00:0c:29:7d:1e:12  10     full
0       0:04:32:28  290      1
0    ge0/3      10.0.20.15/24     Up      Up      null   service    1500  00:0c:29:7d:1e:1c  10     full
0       0:04:32:28  290      2
0    ge0/6      57.0.1.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:3a  10     full
0       0:04:32:28  290      2
0    ge0/7      10.0.100.15/24    Up      Up      null   service    1500  00:0c:29:7d:1e:44  10     full
0       0:04:32:28  300      2
0    system     172.16.255.15/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:04:32:27  0        0
1    ge0/4      10.20.24.15/24    Up      Up      null   service    1500  00:0c:29:7d:1e:26  10     full
0       0:04:32:18  2015     1731
1    ge0/5      56.0.1.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:30  10     full
0       0:04:32:18  290      3
1    ge0/6.2    10.2.2.3/24       Up      Up      vlan   service    1490  00:0c:29:7d:1e:3a  10     full
0       0:04:32:18  0        16335
1    ge0/6.3    10.2.3.5/24       Up      Up      vlan   service    1496  00:0c:29:7d:1e:3a  10     full
0       0:04:32:18  0        16335
512  eth0       10.0.1.15/24      Up      Up      null   service    1500  00:50:56:00:01:0f  1000   full
0       0:04:32:21  3224     1950
```

Configure Loopback Interfaces

You can configure loopback interfaces in any VPN. Use the interface name format **loopback** *string* , where *string* can be any alphanumeric value and can include underscores (_) and hyphens (–). The total interface name, including the string "loopback", can be a maximum of 16 characters long. (Note that because of the flexibility of interface naming in the CLI, the interfaces **lo0** and **loopback0** are parsed as different strings and as such are not interchangeable. For the CLI to recognize as interface as a loopback interface, its name must start with the full string **loopback** .)

One special use of loopback interfaces is to configure data traffic exchange across private WANs, such as MPLS or metro Ethernet networks. To allow a vEdge router that is behind a private network to communicate directly over the private WAN with other vEdge routers, you direct data traffic to a loopback interface that is configured as a tunnel interface rather than to an actual physical WAN interface. For more information, see *Configure Data Traffic Exchange across Private WANs* in the Configuring Segmentation (VPNs) article.

Configure GRE Interfaces and Advertise Services to Them

When a service, such as a firewall, is available on a device that supports only GRE tunnels, you can configure a GRE tunnel on the vEdge router to connect to the remote device. You then advertise that the service is available via a GRE tunnel, and you direct the appropriate traffic to the tunnel either by creating centralized data policy or by configuring GRE-specific static routes.

You create a GRE tunnel by configuring a GRE interface. GRE interfaces are logical interfaces, and you configure them just like any other physical interface. Because a GRE interface is a logical interface, you must bind it to a physical interface, as described below.

To configure a GRE tunnel interface to a remote device that is reachable through a transport network, configure the tunnel in VPN 0:

```
vEdge(config)# vpn 0 interface grenumber
vEdge(config-interface-gre)# (tunnel-source ip-address | tunnel-source-interface interface-name)
vEdge(config-interface-gre)# tunnel-destination ip-address
vEdge(config-interface-gre)# no shutdown
```

The GRE interface has a name in the format **gre** *number* , where *number* can be from 1 through 255.

To configure the source of the GRE tunnel on the local device, you can specify either the IP address of the physical interface (in the **tunnel-source** command) or the name of the physical interface (in the **tunnel-source-interface** command). Ensure that the physical interface is configured in the same VPN in which the GRE interface is located.

To configure the destination of the GRE tunnel, specify the IP address of the remote device in the **tunnel-destination** command.

The combination of a source address (or source interface name) and a destination address defines a single GRE tunnel. Only one GRE tunnel can exist that uses a specific source address (or interface name) and destination address pair.

You can optionally configure an IP address for the GRE tunnel itself:

```
vEdge(config-interface-gre)# ip address ip-address
```

Because GRE tunnels are stateless, the only way for the local router to determine whether the remote end of the tunnel is up, is to periodically send keepalive messages over the tunnel. The keepalive packets are looped back to the sender, and receipt of these packets by the local router indicates that the remote GRE device is up. By default, the GRE interface sends keepalive packets every 10 seconds, and if it receives no response, retries 3 times before declaring the remote device to be down. You can modify these default values with the **keepalive** command:

```
vEdge(config-interface-gre)# keepalive seconds retries
```

The keepalive interval can be from 0 through 65535 seconds, and the number of retries can be from 0 through 255. If you configure an IP address for the GRE interface, that IP address generates the keepalive messages.

If the vEdge router sits behind a NAT and you have configured GRE encapsulation, you must disable keepalives, with a **keepalive 0 0** command. (Note that you cannot disable keepalives by issuing a **no keepalive** command. This command returns the keepalive to its default settings of sending a keepalive packet every 10 seconds and retrying 3 times before declaring the remote device down.)

For GRE interfaces, you can configure only the following additional interface properties:

```
vEdge(config-interface-gre)# access-list acl-name
vEdge(config-interface-gre)# block-non-source-ip
vEdge(config-interface-gre)# clear-dont-fragment
vEdge(config-interface-gre)# description text
vEdge(config-interface-gre)# mtu bytes
vEdge(config-interface-gre)# policer policer-name
vEdge(config-interface-gre)# rewrite-rule rule-name
vEdge(config-interface-gre)# tcp-mss-adjust
```

GRE interfaces do not support cFlowd traffic monitoring.

You can configure one or two GRE interfaces per service. When you configure two, the first interface is the primary GRE tunnel, and the second is the backup tunnel. All packets are sent only to the primary tunnel. If that tunnel fails, all packets are then sent to the secondary tunnel. If the primary tunnel comes back up, all traffic is moved back to the primary GRE tunnel.

You direct data traffic from the service VPN to the GRE tunnel in one of two ways: either with a GRE-specific static route or with a centralized data policy.

To create a GRE-specific static route in the service VPN (a VPN other than VPN 0 or VPN 512), use the **ip gre-route** command:

```
vEdge(config-vpn)# ip gre-route prefix vpn 0 interface grenumber [grenumber2]
```

This GRE-specific static route directs traffic from the specified prefix to the primary GRE interface, and optionally to the secondary GRE interface, in VPN 0. The OMP administrative distance of a GRE-specific static route is 5, and that for a regular static route (configured with the **ip route** command) is 0. For more information, see Unicast Overlay Routing Overview .

To direct the data traffic to the GRE tunnel using a centralized data policy is a two-part process: you advertise the service in the service VPN, and then you create a centralized data policy on the vSmart controller to forward matching traffic to that service.

To advertise the service, include the **service** command in the service VPN (a VPN other than VPN 0 or VPN 512):

```
vEdge(config-vpn)# service service-name interface grenumber [grenumber2]
```

The service name can be **FW** , **IDP** , **IDS** , or **TE** , or a custom service name **netsvc1** through **netsvc4** . The interface is the GRE interface in VPN 0 that is used to reach the service. If you have configured a primary and a backup GRE tunnel, list the two GRE interfaces (**gre**

number1 **gre** *number2*) in the **service** command. Once you have configured a service as reachable a the GRE interface, you cannot delete the GRE interface from the configuration. To delete the GRE interface, you must first delete the service. You can, however, reconfigure the service itself, by modifying the **service** command.

Then, create a data policy on the vSmart controller that applies to the service VPN. In the action portion of the data policy, you must explicitly configure the policy to service the packets destined for the GRE tunnel. To do this, include the **local** option in the **set service** command:

```
vSmart(config-policy-data-policy-vpn-list-vpn-sequence)# action accept
vSmart(config-action-accept)# set service service-name local
```

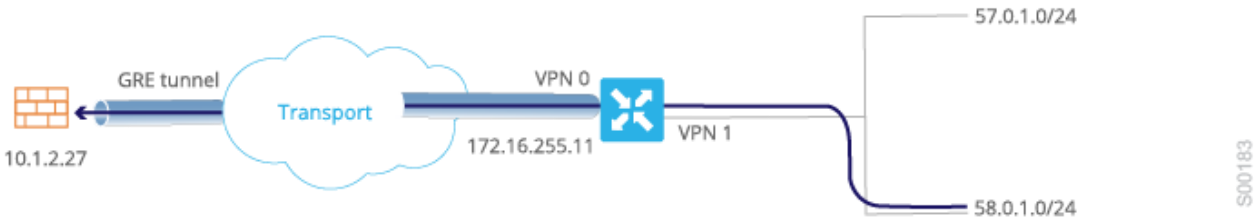
If the GRE tunnel used to reach the service is down, packet routing falls back to using standard routing. To drop packets when a GRE tunnel to the service is unreachable, add the **restrict** option:

```
vSmart(config-policy-data-policy-vpn-list-vpn-sequence)# action accept
vSmart(config-action-accept)# set service service-name local restrict
```

To track GRE tunnels and their traffic, use the following commands:

- **show interface** —List data traffic transmitted and received on GRE tunnels.
- **show tunnel gre-keepalives** —List GRE keepalive traffic transmitted and received on GRE tunnels.
- **show tunnel statistics** —List both data and keepalive traffic transmitted and received on GRE tunnels.

The following figure illustrates an example of configuring a GRE tunnel in VPN 0, to allow traffic to be redirected to a service that is not located at the same site as the vEdge router. In this example, local traffic is directed to the GRE tunnel using a centralized data policy, which is configured on the vSmart controller.



The configuration looks like this:

```
vEdge# show running-config vpn 0
vpn 0
  interface gre1
    ip address 172.16.111.11/24
    keepalive 60 10
    tunnel-source 172.16.255.11
    tunnel-destination 10.1.2.27
    no shutdown
  !
!
vEdge# show running-config vpn 1 service
vpn 1
  service FW interface gre1

vSmart# show running-config policy
policy
  lists
    prefix-list for-firewall
      ip-prefix 58.0.1.0/24
    site-list my-site
      site-id 100
    vpn-list for-vpn-1
```

```
      vpn 1
  data-policy to-gre-tunnel
    vpn-list for-vpn-1
      sequence 10
        match
          source-data-prefix-list for-firewall
        action accept
          set service FW local
apply-policy site-list my-site
  data-policy to-gre-tunnel from-service
```

Here is an example of the same configuring using a GRE-specific static route to direct data traffic from VPN 1 into the GRE tunnels:

```
vEdge# show running-config
vpn 0
  interface gre1
    ip address 172.16.111.11/24
    keepalive 60 10
    tunnel-source 172.16.255.11
    tunnel-destination 10.1.2.27
    no shutdown
  !
!
vpn 1
  ip gre-route 58.0.1.0/24 vpn 0 interface gre1
```

The **show interface** command displays the GRE interface in VPN 0:

```
vEdge# show interface vpn 0
                                  IF      IF
TCP
                                  ADMIN   OPER    ENCAP                                    SPEED
MSS                 RX       TX
VPN  INTERFACE  IP ADDRESS        STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR            MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
----------------------------------------
0    gre1       172.16.111.11/24  Up      Down    null   service    1500  0a:00:05:0b:00:00  -      -
1420    -          0        0
0    ge0/1      10.0.26.11/24     Up      Up      null   service    1500  00:0c:29:ab:b7:62  10     full
1420    0:03:35:14  89       5
0    ge0/2      10.0.5.11/24      Up      Up      null   transport  1500  00:0c:29:ab:b7:6c  10     full
1420    0:03:35:14  9353     18563
0    ge0/3      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:76  10     full
1420    0:03:57:52  99       0
0    ge0/4      10.0.7.11/24      Up      Up      null   service    1500  00:0c:29:ab:b7:80  10     full
1420    0:03:35:14  89       5
0    ge0/5      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:8a  10     full
1420    0:03:57:52  97       0
0    ge0/6      -                 Down    Up      null   service    1500  00:0c:29:ab:b7:94  10     full
1420    0:03:57:52  85       0
0    ge0/7      10.0.100.11/24    Up      Up      null   service    1500  00:0c:29:ab:b7:9e  10     full
1420    0:03:56:30  3146     2402
0    system     172.16.255.11/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
1420    0:03:34:15  0        0
```

You can also view the GRE tunnel information:

```
vEdge# show tunnel gre-keepalives
                                                          REMOTE   REMOTE
     IF                              ADMIN  OPER   KA      TX       RX       TX       RX       TX
RX
VPN  NAME  SOURCE IP  DEST IP         STATE  STATE  ENABLED  PACKETS  PACKETS  PACKETS  PACKETS  ERRORS
ERRORS  TRANSITIONS
-------------------------------------------------------------------------------------------------------
------------------
```

```
0    gre1  10.0.5.11  10.1.2.27       up     down   true     0        0        442      0        0
0       0

vEdge# show tunnel statistics
tunnel statistics gre 10.0.5.11 10.1.2.27 0 0
 tunnel-mtu     1460
 tx_pkts        451
 tx_octets      54120
 rx_pkts        0
 rx_octets      0
 tcp-mss-adjust 1380
```

Set the Interface Speed

When a vEdge router comes up, the Viptela software autodetects the SFPs present in the router and sets the interface speed accordingly. The software then negotiates the interface speed with the device at the remote end of the connection to establish the actual speed of the interface. To display the hardware present in the router, use the **show hardware inventory** command:

```
vEdge# show hardware inventory

             HW
             DEV
HW TYPE      INDEX  VERSION  PART NUMBER       SERIAL NUMBER     DESCRIPTION
-------------------------------------------------------------------------------------------------------
--------------
Chassis      0      3.1      vEdge-1000        11OD145130001     vEdge-1000
CPU          0      None     None              None              Quad-Core Octeon-II
DRAM         0      None     None              None              2048 MB DDR3
Flash        0      None     None              None              nor Flash - 16.00 MB
eMMC         0      None     None              None              eMMC - 7.31 GB
PIM          0      None     ge-fixed-8        None              8x 1GE Fixed Module
Transceiver  0      A        FCLF-8521-3       PQD3FHL           Port 0/0, Type 0x8 (Copper), Vendor
FINISAR CORP.
Transceiver  1      PB       1GBT-SFP05        0000000687        Port 0/1, Type 0x8 (Copper), Vendor BEL-
FUSE
FanTray      0      None     None              None              Fixed Fan Tray - 2 Fans
```

To display the actual speed of each interface, use the **show interface** command. Here, interface **ge0/0** , which connects to the WAN cloud, is running at 1000 Mbps (1Gbps; it is the 1GE PIM highlighted in the output above), and interface **ge0/1** , which connects to a device at the local site, has negotiated a speed of 100 Mbps.

```
vEdge# show interface

                                IF      IF
TCP
                                ADMIN   OPER    ENCAP                                   SPEED
MSS                 RX       TX
VPN  INTERFACE  IP ADDRESS      STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR            MBPS   DUPLEX
ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
---------------------------------------
0    ge0/0      192.168.1.4/24  Up      Up      null   transport  1500  00:0c:bd:05:f0:83  1000   full
1300    0:06:10:59  2176305  2168760
0    ge0/2      -               Down    Down    null   service    1500  00:0c:bd:05:f0:81  -      -
0       -           0        0
0    ge0/3      -               Down    Down    null   service    1500  00:0c:bd:05:f0:82  -      -
0       -           0        0
0    ge0/4      -               Down    Down    null   service    1500  00:0c:bd:05:f0:87  -      -
0       -           0        0
0    ge0/5      -               Down    Down    null   service    1500  00:0c:bd:05:f0:88  -      -
0       -           0        0
0    ge0/6      -               Down    Down    null   service    1500  00:0c:bd:05:f0:85  -      -
```

```
0       -           0        0
0    ge0/7      -               Down    Down    null   service    1500  00:0c:bd:05:f0:86  -      -
0       -           0        0
0    system     1.1.1.1/32      Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:06:11:15  0        0
1    ge0/1      10.192.1.1/28   Up      Up      null   service    1500  00:0c:bd:05:f0:84  100    full
0       0:06:10:59  87       67
1    loopback1  1.1.1.1/32      Up      Up      null   service    1500  00:00:00:00:00:00  10     full
0       0:06:10:59  0        0
2    loopback0  10.192.1.2/32   Up      Up      null   service    1500  00:00:00:00:00:00  10     full
0       0:06:10:59  0        0
512  mgmt0      -               Up      Down    null   mgmt       1500  00:0c:bd:05:f0:80  -      -
0       -           0        0
```

For non-physical interfaces, such as those for the system IP address and loopback interfaces, the interface speed is set by default to 10 Mbps.

To override the speed negotiated by the two devices on the interface, disable autonegotiation and configure the desired speed:

```
vEdge(config-vpn)# interface interface-name no autonegotiate
vEdge(config-vpn)# interface interface-name speed (10 | 100)
```

For vSmart controllers and vManage NMS systems, the initial interface speeds are 1000 Mbps, and the operating speed is negotiated with the device at the remote end of the interface.

Set the Interface MTU

By default, all interfaces have an MTU of 1500 bytes. You can modify this on an interface:

```
Viptela(config-vpn)# interface interface-name mtu bytes
```

The MTU can range from 576 through 2000 bytes.

To display an interface's MTU, use the **show interface** command.

For vBond, vManage, and vSmart devices, you can configure interfaces to use ICMP to perform path MTU (PMTU) discovery. When PMTU discovery is enabled, the device to automatically negotiates the largest MTU size that the interface supports in an attempt to minimize or eliminate packet fragmentation:

```
Viptela(config-vpn)# interface interface-name pmtu
```

On vEdge routers, the Viptela BFD software automatically performs PMTU discovery on each transport connection (that is, for each TLOC, or color). BFD PMTU discovery is enabled by default, and it is recommended that you use it and not disable it. To explicitly configure BFD to perform PMTU discovery, use the **bfd color pmtu-discovery** configuration command. However, you can choose to instead use ICMP to perform PMTU discovery:

```
vEdge(config-vpn)# interface interface-name pmtu
```

BFD is a data plane protocol and so does not run on vBond, vManage, and vSmart devices.

Configure Static ARP Table Entries

By default, vEdge routers respond to ARP requests only if the destination address of the request is on the local network. You can configure static ARP entries on a Gigabit Ethernet interface in any VPN to allow the router to respond to ARP requests even if the destination address of the request is not local to the incoming interface. The ARP entry maps an IP address to a MAC address.

```
Viptela(config-vpn)# interface interface-name arp ip ip-address mac mac-address
```

Monitoring Bandwidth on a Transport Circuit

You can monitor the bandwidth usage on a transport circuit, to determine how the bandwidth usage is trending. If the bandwidth usage starts approaching a maximum value, you can configure the software to send a notification. Notifications are sent as Netconf notifications, which are sent to the vManage NMS, SNMP traps, and syslog messages. You might want to enable this feature for bandwidth monitoring, such as when you are doing capacity planning for a circuit or when you are gathering trending information about bandwidth utilization. You might also enable this feature to receive alerts regarding bandwidth usage, such as if you need to determine when a transport interface is becoming so saturated with traffic that a customer's traffic is impacted, or when customers have a pay-per-use plan, as might be the case with LTE transport.

To monitor interface bandwidth, you configure the maximum bandwidth for traffic received and transmitted on a transport circuit. The maximum bandwidth is typically the bandwidth that has been negotiated with the circuit provider. When bandwidth usage exceeds 85 percent of the configured value for either received or transmitted traffic, a notification, in the form of an SNMP trap, is generated. Specifically, interface traffic is sampled every 10 seconds. If the received or transmitted bandwidth exceeds 85 percent of the configured value in 85 percent of the sampled intervals in a continuous 5-minute period, an SNMP trap is generated. After the first trap is generated, sampling continues at the same frequency, but notifications are rate-limited to once per hour. A second trap is sent (and subsequent traps are sent) if the bandwidth exceeds 85 percent of the value in 85 percent of the 10-second sampling intervals over the next 1-hour period. If, after 1 hour, another trap is not sent, the notification interval reverts to 5 minutes.

You can monitor transport circuit bandwidth on vEdge routers and on vManage NMSs.

To generate notifications when the bandwidth of traffic received on a physical interface exceeds 85 percent of a specific bandwidth, configure the downstream bandwidth:

```
vEdge/vManage(config)# vpn vpn-id interface interface-name bandwidth-downstream kbps
```

To generate notifications when the bandwidth of traffic transmitted on a physical interface exceeds 85 percent of a specific bandwidth, configure the upstream bandwidth:

```
vEdge/vManage(config)# vpn vpn-id interface interface-name bandwidth-upstream kbps
```

In both configuration commands, the bandwidth can be from 1 through 2147483647 ($2^{32} / 2$) – 1 kbps.

To display the configured bandwidths, look at the bandwidth-downstream and bandwidth-upstream fields in the output of the **show interface detail** command. The rx-kbps and tx-kbps fields in this command shows the current bandwidth usage on the interface.

Additional Information

Configuring Cellular Interfaces
Configuring DHCP
Configuring IKE-Enabled IPsec Tunnels
Configuring PPPoE
Configuring Segmentation (VPNs)
Configuring VRRP
Configuring WLAN Interfaces
interface
Network Interface Configuration Examples
show interface
show ipv6 interface

Configuring Cellular Interfaces

To enable LTE connectivity, you configure cellular interfaces on vEdge routers that have a cellular module. The cellular module provides wireless connectivity over a service provider's cellular network. One use case is to provide wireless connectivity for branch offices.

A cellular network is commonly used as a backup WAN link, to provide network connectivity if all the wired WAN tunnel interfaces on the router become unavailable. You can also use a cellular network as the primary WAN link for a branch office, depending on usage patterns within the branch office and the data rates supported by the core of the service provider's cellular network.

When you configure a cellular interface on a vEdge router, you can connect the router to the Internet or other WAN simply by plugging in the router's power cable. The vEdge router then automatically begins the process of joining the overlay network, by contacting and authenticating with vBond orchestrators, vSmart controllers, and vManage NMSs.

vEdge routers support LTE and CDMA radio access technology (RAT) types.

Configure a Cellular Interface

To use the CLI to configure a cellular interface on a vEdge router that has a cellular module:

1. Create a cellular profile:
   vEdge(config)# **cellular cellular** *number*
   vEdge(config-cellular)# **profile** *profile-id*
   Each router has only one LTE module, so *number* must be 0. The profile identifier can be a value from 1 through 15.

2. If your ISP requires that you configure profile properties, configure one or more of the following:
   vEdge(config-profile)# **apn** *name*
   vEdge(config-profile)# **auth** *auth-method*
   vEdge(config-profile)# **ip-addr** *ip-address*
   vEdge(config-profile)# **name** *name*
   vEdge(config-profile)# **pdn-type** *type*
   vEdge(config-profile)# **primary-dns** *ip-address*
   vEdge(config-profile)# **secondary-dns** *ip-address*
   vEdge(config-profile)# **user-name** *username*
   vEdge(config-profile)# **user-pass** *password*

   Note: If you want to remove a property from the cellular profile, delete the profile entirely from the configuration, and create it again with only the required parameters.

3. Create the cellular interface:
   vEdge(config)# **vpn 0 interface cellular0**

4. Enable the cellular interface:
   vEdge(confg-interface)# **no shutdown**

5. For cellular interfaces, you must use a DHCP client to dynamically configure the IP address. This is the default option. To explicitly configure this:
   vEdge(config-interface)# **ip dhcp-client** [**dhcp-distance** *number*]
   number is the administrative distance of routes learned from a DHCP server. You can configure it to a value from 1 through 255.

6. Associate the cellular profile with the cellular interface:
   vEdge(config-interface)# **profile** *profile-id*
   The profile identifier is the number you configured in Step 1.

7. Set the interface MTU:
   vEdge(config-interface)# **mtu** *bytes*
   The MTU can be 1428 bytes or smaller.

8. By default, the radio access technology (RAT) type is LTE. For 2G/3G networks, change it to CDMA:
   vEdge(config-interface)# **technology cdma**
   If you are using the interface for ZTP, change the technology to **auto** :
   vEdge(config-interface)# **technology auto**

9. Configure any other desired interface properties.

10. Create a tunnel interface on the cellular interface:
    vEdge(config-interface)# **tunnel-interface**
    vEdge(config-tunnel-interface)# **color** *color*
    vEdge(config-tunnel-interface)# **encapsulation** (**gre** | **ipsec**)

11. By default, the tunnel interface associated with a cellular interface is not considered to be the circuit of last resort. To allow the tunnel to be the circuit of last resort:
    vEdge(config-tunnel-interface)# **last-resort-circuit**
    When the interface is configured as a circuit of last resort, the cellular modem becomes dormant and no traffic is sent over the circuit. However, the cellular modem is kept in online mode so that the modem radio can be monitored at all times and to allow for faster switchover in the case the tunnel interface needs to be used as the last resort.
    By default, there is a delay of 7 seconds before switching back to the primary tunnel interface from a circuit of last resort. This delay is to ensure that the primary interface is once again fully operational and is not still flapping.

12. To minimize the amount of control plane keepalive traffic on the cellular interface, increase the Hello packet interval and tolerance on the tunnel interface:
    vEdge(config-tunnel-interface)# **hello-interval** *milliseconds*
    vEdge(config-tunnel-interface)# **hello-tolerance** *seconds*
    The default hello interval is 1000 milliseconds, and it can be a time in the range 100 through 600000 milliseconds (10 minutes). The default hello tolerance is 12 seconds, and it can be a time in the range 12 through 600 seconds (10 minutes). To reduce outgoing control packets on a TLOC, it is recommended that on the tunnel interface you set the hello interface to 60000 milliseconds (10 minutes) and the hello tolerance to 600 seconds (10 minutes) and include the **no track-transport** disable regular checking of the DTLS connection between the router and the vBond orchestrator.
    For a tunnel connection between a vEdge router and any controller device, the tunnel uses the hello interval and tolerance times configured on the router. This choice is made to minimize the amount traffic sent over the tunnel, to allow for situations where the cost of a link is a function of the amount of traffic traversing the link. The hello interval and tolerance times are chosen separately for each tunnel between a vEdge router and a controller device.
    Another step taken to minimize the amount of control plane traffic is to not send or receive OMP control traffic over a cellular interface when other interfaces are available. This behavior is inherent in the software and is not configurable.

13. If the router has two or more cellular interfaces, you can minimize the amount of traffic between the vManage NMS and the cellular interfaces by setting one of the interfaces to be the preferred one to use when sending updates to the vManage NMS and receiving configurations from the vManage NMS:
    vEdge(config-tunnel-interface)# **vmanage-connection-preference** *number*
    The preference can be a value from 0 through 8. The default preference is 5. To have a tunnel interface never connect to the vManage NMS, set the number to 0. At least one tunnel interface on the router must have a nonzero vManage connection preference.

14. Configure any other desired tunnel interface properties.

15. To minimize the amount of data plane keepalive traffic on the cellular interface, increate the BFD Hello packet interval:
    vEdge(bfd-color-lte)# **hello-interval** *milliseconds*
    The default hello interval is 1000 milliseconds (1 second), and it can be a time in the range 100 through 300000 milliseconds (5 minutes).

To determine the status of the cellular hardware, use the **show cellular status** command.

To determine whether a vEdge has a cellular module, use the **show hardware inventory** command.

To determine whether a cellular interface is configured as a last-resort circuit, use the **show control affinity config** and **show control local-properties** commands.

Note: When you activate the configuration on a router with cellular interfaces, the primary interfaces (that is, those interfaces not configured as circuits of last resort) and the circuit of last resort come up. In this process, all the interfaces begin the process of establishing control and BFD connections. When one or more of the primary interfaces establishes a TLOC connection, the circuit of last resort shuts itself down because it is not needed. During this shutdown process, the circuit of last resort triggers a BFD TLOC Down alarm and a Control TLOC Down alarm on the vEdge router. These two alarms are cleared only when all the primary interfaces lose their BFD connections to remote nodes and the circuit of last resort activates itself. This generation and clearing of alarms is expected behavior.

Best Practices for Configuring Cellular Interfaces

Cellular technology on Viptela devices can be used in a number of ways:

- Circuit of last resort—You can use a cellular interface as a backup circuit on a vEdge router. Such a circuit is activated only if all transport links on the router fail. In this mode the radio interface is turned off, and no control or data connections exist over the cellular interface.

To configure an cellular interface to be a circuit of last resort, include the **last-resort-circuit** command when you configure the cellular interface's tunnel interface.

- Active circuit—You can choose to use a cellular interface as an active circuit, perhaps because it is the only last-mile circuit or to always keep the cellular interface active so that you can measure the performance of the circuit. In this scenario the amount of bandwidth utilized to maintain control and data connections over the cellular interface can become a concern. Here are some best practices to minimize bandwidth usage over a cellular interface:

  – When the vEdge router with cellular interface is deployed as a spoke and data tunnels are established in a hub-and-spoke manner, you can configure the cellular interface as a low-bandwidth interface. To do this, include the **low-bandwidth-link** command when you configure the cellular interface's tunnel interface. When the cellular interface is operating as a low-bandwidth interface, the vEdge router spoke site is able to synchronize all outgoing control packets. The spoke site can also proactively ensure that no control traffic, except for routing updates, is generated from one of the remote hub nodes. Routing updates continue to be sent, because they are considered to be critical updates.

  – Increase control packet timers—To minimize control traffic on a cellular interface, you can decrease how often protocol update messages are sent on the interface. OMP sends Update packets every second, by default. You can increase this interval to a maximum of 65535 seconds (about 18 hours) by including the **omp timers advertisement-interval** configuration command. BFD sends Hello packets every second, by default. You can increase this interval to a maximum of 5 minutes (300000 milliseconds) by including the **bfd color hello-interval** configuration command. (Note that you specify the OMP Update packet interval in seconds and the BFD Hello packet interval in milliseconds.)

  – Prioritize vManage control traffic over a non-cellular interface—When a vEdge router has both cellular and non-celluar transport interfaces, by default, the router chooses one of the interfaces to use to exchange control traffic with the vManage NMS. You can configure the router to never use the cellular interface to exchange traffic with the NMS, or you can configure a lower preference for using the cellular interface for this traffic. You configure the preference by including the **vmanage-connection-preference** command when configuring the tunnel interface. By default, all tunnel interface have a vManage connection preference value of 5. The value can range from 0 through 8, where a higher value is more preferred. A tunnel with a preference value of 0 can never exchange control traffic with the vManage NMS.

Note: At least one tunnel interface on the vEdge router must have a non-0 vManage connection preference value. Otherwise, the router has no control connections.

Additional Information

Configuring DHCP
Configuring Network Interfaces
Configuring PPPoE
Configuring VRRP
Configuring WLAN Interfaces
Software Caveats in Release 18.1 Release Notes
Troubleshoot Cellular Interfaces

Configuring DHCP

When you configure a tunnel interface on a vEdge router, a number of services are enabled by default on that interface, including DHCP.

A vEdge router can act as a DHCP server for the service-side network to which it is connected, and it can also act as a DHCP helper, forwarding requests for IP addresses from devices in the service-side network to a DHCP server that is in a different subnet on the service side of the vEdge router.

Enable DHCP on the WAN Interface

On a vEdge router's WAN interface—the interface configured as a tunnel interface in VPN 0, the transport VPN—DHCP is enabled by default. You can see this by using the **details** filter with the **show running-config** command. This command also shows that the DNS and ICMP services are enabled by default.

```
vm1# show running-config vpn 0 interface ge0/2 tunnel-interface | details
vpn 0
 interface ge0/2
  tunnel-interface
   encapsulation ipsec weight 1
   color lte
   control-connections
   carrier             default
   no allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service ospf
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
 !
!
```

Enabling DHCP on the router's WAN interface allows the device that actually connects the router to the transport network (such as a DSL router) to dynamically assign a DHCP address to the vEdge router. The DHCP service in VPN 0 affects the transport-side network.

Have a vEdge Router Be a DHCP Server

One or more service-side interfaces on vEdge router can act as a DHCP server, assigning IP addresses to hosts in the service-side network. To do this, configure this function on the interface that connects the vEdge router to the local site's network. At a minimum, you must configure the pool of IP addresses available for assigning to hosts:

```
vEdge(config-vpn)# interface geslot/port dhcp-server address-pool ip-address/prefix
vEdge(config-dhcp-server)#
```

You can exclude IP addresses that fall within the range of the DHCP address pool:

```
vEdge(config-dhcp-server)# exclude ip-address
```

To specify multiple individual addresses, list them in a single **exclude** command, separated by a space (for example, **exclude 1.1.1.1 2.2.2.2 3.3.3.3**). To specify a range of addresses, separate them with a hyphen (for example, **exclude 1.1.1.1-1.1.1.10**).

You can also statically assign IP addresses to a host:

```
vEdge(config-dhcp-server)# static-lease mac-address ip ip-address
```

By default, the DHCP server on a single interface can assign 254 DHCP leases, and each lease is valid for 24 hours. The offer of an IP address is valid indefinitely, until that DHCP server runs out of addresses to offer. You can modify these values:

```
vEdge(config-dhcp-server)# max-leases number
vEdge(config-dhcp-server)# lease-time seconds
vEdge(config-dhcp-server)# offer-time seconds
```

These values can range from 0 through ($2^{32} - 1$).

The Viptela software supports DHCP server options that allow you to configure the IP addresses of a default gateway, DNS server, and TFTP server in the service-side network and the network mask of the service-side network:

```
vEdge(config-dhcp-server)# options default-gateway ip-address
vEdge(config-dhcp-server)# options dns-servers ip-address
vEdge(config-dhcp-server)# options domain-name domain-name
vEdge(config-dhcp-server)# options interface-mtu mtu
vEdge(config-dhcp-server)# options tftp-servers ip-address
```

Have a vEdge Router Be a DHCP Helper

One or more service-side interfaces on a vEdge router can be a DHCP helper. With this configuration, the interface forwards any broadcast BOOTP DHCP requests that it receives from hosts on the service-side network to the DHCP server or servers specified by the configured IP helper address (or addresses) and returns the assigned IP address to the requester.

When the DHCP server at the vEdge router's local site is on a different segment than the devices connected to the vEdge router or than the vEdge router itself. When configured as a DHCP helper, the vEdge interface forwards any broadcast BOOTP DHCP requests that it receives to the DHCP server specified by the configured IP helper address.

To configure an interface as a DHCP helper, configure the IP address of the DHCP server on the interface that connects to the local site's network:

```
vEdge(config-vpn)# interface geslot/port dhcp-helper ip-address
```

You can configure up to four IP addresses, and you must enter the addresses in a single **dhcp-helper** command.

In Releases 17.2.2 and later, you can configure up to eight IP address. You must enter all the addresses in a single **dhcp-helper** command.

Additional Information

Configuring Cellular Interfaces
Configuring Network Interfaces
Configuring PPPoE
Configuring VRRP
Configuring WLAN Interfaces

Configuring PPPoE

The Point-to-Point Protocol over Ethernet (PPPoE) connects multiple users over an Ethernet local area network to a remote site through common customer premises equipment. PPPoE is commonly used in a broadband aggregation, such as by digital subscriber line (DSL). PPPoE provides authentication with the CHAP or PAP protocol. In the Viptela overlay network, vEdge routers can run the PPPoE client. The PPPoE server component is not supported.

To configure PPPoE client on a vEdge router, you create a PPP logical interface and link it to a physical interface. The PPPoE connection comes up when the physical interface comes up. You can link a PPP interface to only one physical interface on a vEdge router, and you can link a physical interface to only one PPP interface. To enable more than one PPPoE interface on a vEdge router, configure multiple PPP interfaces.

You configure quality of service (QoS) and shaping rate on a PPPoE-enabled physical interface, rather than on the PPP interface.

PPPoE-enabled physical interfaces do not support:

- 802.1Q
- Subinterfaces
- NAT, PMTU, and tunnel interfaces. These are configured on the PPP interface and therefore not available on PPPoE-enabled interfaces.

The Viptela implementation of PPPoE does not support the Compression Control Protocol (CCP) options, as defined in RFC 1962.

Configure PPPoE from vManage Templates

To use vManage templates to configure PPPoE on vEdge routers, you create three feature templates and one device template:

- Create a VPN-Interface-PPP feature template to configure PPP parameters for the PPP virtual interface.
- Create a VPN-Interface-PPP-Ethernet feature template to configure a PPPoE-enabled interface.

- Optionally, create a VPN feature template to modify the default configuration of VPN 0.
- Create a device template that incorporates the VPN-Interface-PPP, VPN-Interface-PPP-Ethernet, and VPN feature templates.

To create a VPN-Interface-PPP feature template to configure PPP parameters for the PPP virtual interface:

Parameter Field	Procedure
Template Name	Enter a name for the template. It can be up to 128 alphanumeric characters.
Description	Enter a description for the template. It can be up to 2048 alphanumeric characters.
Shutdown	Click No to enable the PPP virtual interface.
Interface Name	Enter the number of the PPP interface. It can be from 1 through 31.
Description (optional)	Enter a description for the PPP virtual interface.
Authentication Protocol	Select either CHAP or PAP to configure one authentication protocol, or select PAP and CHAP to configure both. For CHAP, enter the hostname and password provided by your ISP. For PAP, enter the username and password provided by your ISP. If you are configuring both PAP and CHAP, to use the same username and password for both, click Same Credentials for PAP and CHAP.
AC Name (optional)	Select the PPP tab, and in the AC Name field, enter the name of the the name of the access concentrator used by PPPoE to route connections to the Internet.
IP MTU	Click the Advanced tab, and In the IP MTU field, ensure that the IP MTU is at least 8 bytes less than the MTU on the physical interface. The maximum MTU for a PPP interface is 1492 bytes. If the PPPoE server does not specify a maximum receive unit (MRU), the MTU value for the PPP interface is used as the MRU.
Save	Click Save to save the feature template.

1. In vManage NMS, select the Configuration ► Templates screen.
2. From the Templates title bar, select Feature.
3. Click Add Template.
4. In the left pane, select vEdge Cloud or a router model.
5. In the right pane, select the VPN-Interface-PPP template.
6. In the template, configure the following parameters:

To create a VPN-Interface-PPP-Ethernet feature template to enable the PPPoE client on the physical interfaces:

1. In the vManage NMS, select the Configuration ► Templates screen.
2. From the Templates title bar, select Feature.
3. Click Add Template.
4. In the left pane, select vEdge Cloud or a router model.
5. In the right pane, select the VPN-Interface-PPP-Ethernet template.
6. In the template, configure the following parameters:

Parameter Field	Procedure
Template Name	Enter a name for the template. It can be up to 128 alphanumeric characters.

Description	Enter a description for the template. It can be up to 2048 alphanumeric characters.
Shutdown	Click No to enable the PPPoE-enabled interface.
Interface Name	Enter the name of the physical interface in VPN 0 to associate with the PPP interface.
Description (optional)	Enter a description for the PPPoE-enabled interface.
IP Confguration	Assign an IP address to the physical interface: To use DHCP, select Dynamic. The default administrative distance of routes learned from DHCP is 1. To configure the IP address directly, enter of the IPv4 address of the interface.
DHCP Helper (optional)	Enter up to four IP addresses for DHCP servers in the network.
Save	Click Save to save the feature template.

To create a VPN feature template to configure the PPPoE-enabled interface in VPN 0, the transport VPN:

1. In the vManage NMS, select the Configuration ► Templates screen.
2. From the Templates title bar, select Feature.
3. Click Add Template.
4. In the left pane, select vEdge Cloud or a router model.
5. In the right pane, select the VPN template.
6. In the template, configure the following parameters:

Parameter Field	Procedure
Template Name	Enter a name for the template. It can be up to 128 alphanumeric characters.
Description	Enter a description for the template. It can be up to 2048 alphanumeric characters.
VPN Identifier	Enter VPN identifier 0.
Name	Enter aname for the VPN.
Other interface parameters	Configure the desired interface properties.
Save	Click Save to save the feature template.

To create a device template that incorporates the VPN-Interface-PPP, VPN-Interface-PPP-Ethernet, and VPN feature templates:

1. In the vManage NMS, select the Configuration ► Templates screen.
2. From the Templates title bar, select Device.
3. Click Create Template, and from the drop-down list select From Feature Template.
4. From the Device Model drop-down, select the type of device for which you are creating the device template. vManage NMS displays the feature templates for the device type you selected. Required templates are indicated with an asterisk (*).
5. Enter a name and description for the device template. These fields are mandatory. The template name cannot contain special characters.
6. In the Transport & Management VPN section, under VPN 0, from the drop-down list of available templates, select the desired feature template. The list of available templates are the ones that you have previously created.

7. In the Additional VPN 0 Templates section to the right of VPN 0, click the plus sign (+) next to VPN Interface PPP.
8. In the VPN-Interface-PPP and VPN-Interface-PPP-Ethernet fields, select the feature templates to use.
9. To configure multiple PPPoE-enabled interfaces in VPN 0, click the plus sign (+) next to Sub-Templates.
10. To include additional feature templates in the device template, in the remaining sections, select the feature templates in turn, and from the drop-down list of available templates, select the desired template. The list of available templates are the ones that you have previously created. Ensure that you select templates for all mandatory feature templates and for any desired optional feature templates.
11. Click Create to create the device template.

To attach a device template to a device:

1. In the vManage NMS, select the Configuration ► Templates screen.
2. From the Templates title bar, select Device.
3. Select a template.
4. Click the More Actions icon to the right of the row and click Attach Device.
5. In the Attach Device window, either search for a device or select a device from the Available Device(s) column to the left.
6. Click the arrow pointing right to move the device to the Selected Device(s) column on the right.
7. Click Attach.

Configure PPPoE from the CLI

To use the CLI to configure PPPoE on vEdge routers:

1. Create a PPP interface. The interface number can be from 1 through 31.
   vEdge(config-vpn)# **interface ppp** *number*
2. Configure an authentication method for PPPoE and authentication credentials. You can configure both CHAP and PAP authentication on the same PPP interface. The software tries both methods and uses the first one that succeeds.
   vEdge(config-interface-ppp)# **ppp authentication chap hostname** *name* **password** *password*
   vEdge(config-interface-ppp)# **ppp authentication pap password** *password* **sent-username** *username*
3. Enable the PPP interface to be operationally up:
   vEdge(config-interface-ppp)# **no shutdown**
4. Configure the MTU of the PPP interface. The maximum MTU for a PPP interface is 1492 bytes. If maximum receive unit (MRU) is not specified by the PPPoE server, the MTU value for the PPP interface is used as the MRU.
   vEdge(config-interface-ppp)# **mtu** *bytes*
5. Configure a tunnel interface for the PPP interface:
   vEdge(config-interface-ppp)# **tunnel-interface color** *color*
6. Optionally, configure the name of the access concentrator used by PPPoE to route connections to the internet:
   vEdge(config-interface-ppp)# **ac-name** *name*
7. Link a physical Gigabit Ethernet interface in VPN 0 to the PPP interface:
   vEdge(config-interface-ge)# **pppoe-client ppp-interface ppp** *number*
8. Enable the physical Gigabit Ethernet interface to be operationally up:
   viptela(config-vpn-interface-ge)# **no shutdown**

Here is an example of a PPPoE configuration:

```
vEdge# show running-config vpn 0
vpn 0
 interface ge0/1
  pppoe-client ppp-interface ppp10
  no shutdown
 !
 interface ppp10
  ppp authentication chap
   hostname branch100@corp.bank.myisp.net
   password $4$OHHjdmsC6M8zj4BgLEFXKw==
  !
  tunnel-interface
   encapsulation ipsec
   color gold
   no allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service ospf
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  mtu      1492
  no shutdown
 !
!
```

To view existing PPP interfaces, use the **show ppp interface** command. For example:

```
vEdge# show ppp interface

              PPPOE      INTERFACE               PRIMARY  SECONDARY
VPN  IFNAME   INTERFACE  IP         GATEWAY IP   DNS      DNS        MTU
-------------------------------------------------------------------------
0    ppp10    ge0/1      11.1.1.1   115.0.1.100  8.8.8.8  8.8.4.4    1150
```

To view PPPoE session information, use the **show pppoe session** command. For example:

```
vEdge# show pppoe session

              SESSION                                          PPP                    SERVICE
VPN  IFNAME   ID       SERVER MAC         LOCAL MAC           INTERFACE  AC NAME     NAME
-------------------------------------------------------------------------------------------
0    ge0/1    1        00:0c:29:2e:20:1a  00:0c:29:be:27:f5   ppp1       branch100   -
0    ge0/3    1        00:0c:29:2e:20:24  00:0c:29:be:27:13   ppp2       branch100   -
```

Additional Information

Configuring Cellular Interfaces
Configuring DHCP
Configuring Network Interfaces
Configuring VRRP
Configuring WLAN Interfaces

Configuring VRRP

The Virtual Router Redundancy Protocol (VRRP) provides redundant gateway service for switches and other IP end stations. In the Viptela software, you configure VRRP on an interface, and typically on a subinterface, within a VPN.

For a VRRP interface to operate, its physical interface must be configured in VPN 0:

```
vEdge(config-vpn-0)# interface ge-slot/port
vEdge(config-interface-ge)# no shutdown
```

For each VRRP interface (or subinterface), you assign an IP address and you place that interface in a VRRP group.

```
vEdge(config-vpn)# interface ge-slot/port.subinterface
vEdge(config-interface-ge)# ip address prefix/length
vEdge(config-interface-ge)# vrrp group-number
```

The group number identifies the virtual router. In a typical VRRP topology, two physical routers are configured to act as a single virtual router, so you configure the same group number on interfaces on both these routers.

For each virtual router ID, you must configure an IP address:

```
vEdge(config-vrrp)# ipv4 ip-address
```

Within each VRRP group, the vEdge router with the higher priority value is elected as primary. By default, each virtual router IP address has a default primary election priority of 100, so the router with the higher IP address is elected primary. You can modify the priority value, setting it to a value from 1 through 254:

```
vEdge(config-vrrp)# priority number
```

The VRRP primary periodically sends advertisement messages, indicating that it is still operating. If backup routers miss three consecutive VRRP advertisements, they assume that the primary is down and elect a new primary. By default, these messages are sent every second. You can change the VRRP advertisement time to be a value from 1 through 3600 seconds:

```
vEdge(config-vrrp)# timer seconds
```

By default, VRRP uses of the state of the interface on which it is running to determine which vEdge router is the primary virtual router. This interface is on the service (LAN) side of the vEdge router. When the interface for the primary goes down, a new VRRP primary virtual router is elected based on the VRRP priority value. Because VRRP runs on a LAN interface, if a vEdge router loses all its WAN control connections, the LAN interface still indicates that it is up even though the router is functionally unable to participate in VRRP. To take WAN side connectivity into account for VRRP, you can configure one of the following:

- Track the Overlay Management Protocol (OMP) session running on the WAN connection when determining the VRRP primary virtual router:

  ```
  vEdge(config-vrrp)# track-omp
  ```

  If all OMP sessions are lost on the primary VRRP router, VRRP elects a new default gateway from among all the gateways that have one or more active OMP sessions even if the gateway chosen has a lower VRRP priority than the current primary. With this option, VRRP failover occurs once the OMP state changes from up to down, which occurs when the OMP hold timer expires. (The default OMP hold timer interval is 60 seconds.) Until the hold timer expires and a new VRRP primary is elected, all overlay traffic is dropped. When the OMP session recovers, the local VRRP interface claims itself as primary even before it learns and installs OMP routes from the vSmart controllers. Until the routers are learned, traffic is also dropped.

- Track both the OMP session and a list of remote prefixes. *list-name* is the name of a prefix list configured with the **policy lists prefix-list** command on the vEdge router:

  ```
  vEdge(config-vrrp)# track-prefix-list list-name
  ```

  If all OMP sessions are lost, VRRP failover occurs as described for the **track-omp** option. In addition, if reachability to all the prefixes in the list is lost, VRRP failover occurs immediately, without waiting for the OMP hold timer to expire, thus minimizing the amount of overlay traffic is dropped while the vEdge routers determine the VRRP primary.

As discussed above, the IEEE 802.1Q protocol adds 4 bytes to each packet's length. Hence, for packets to be transmitted, either increase the MTU size on the physical interface in VPN 0 (the default MTU is 1500 bytes) or decrease the MTU size on the VRRP interface. See the example configuration output below.

Here is an example of configuring VRRP on redundant physical interfaces. For subinterface 2, vEdge1 is configured to act as the primary, and for subinterface 3, vEdge2 acts as the primary.

```
vEdge1# show running-config vpn 1
vpn 1
```

```
 interface ge0/6.2
  ip address 10.2.2.3/24
  mtu      1496
  no shutdown
  vrrp 2
   ipv4 10.2.2.1
   track-prefix-list vrrp-prefix-list1
  !
 !
 interface ge0/6.3
  ip address 10.2.3.5/24
  mtu      1496
  shutdown
  vrrp 3
   ipv4 10.2.3.11
   track-prefix-list vrrp-prefix-list1
  !
 !
!

vEdge2# show running-config vpn 1
vpn 1
 interface ge0/1.2
  ip address 10.2.2.4/24
  mtu      1496
  no shutdown
  vrrp 2
   ipv4 10.2.2.2
   track-prefix-list vrrp-prefix-list2
  !
 !
 interface ge0/1.3
  ip address 10.2.3.6/24
  mtu      1496
  no shutdown
  vrrp 3
   ipv4 10.2.3.12
   track-prefix-list vrrp-prefix-list2
  !
 !
!

vEdge1# show interface vpn 1

                            IF      IF                                                                  TCP
                            ADMIN   OPER    ENCAP  PORT                                 SPEED          MSS
RX       TX
VPN  INTERFACE  IP ADDRESS  STATUS  STATUS  TYPE   TYPE     MTU   HWADDR             MBPS   DUPLEX
ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
----------------------------------
1    ge0/6.2    10.2.2.3/24  Up      Up      vlan   service  1496  00:0c:29:ab:b7:94  10     full    0
0:00:05:52  0        357
1    ge0/6.3    10.2.3.5/24  Down    Down    vlan   service  1496  00:0c:29:ab:b7:94  -      -       0
-           0        0

vEdge1# show vrrp interfaces

                                                                                          MASTER
TRACK   PREFIX
             GROUP  VIRTUAL                                VRRP    OMP    ADVERTISEMENT  DOWN
PREFIX  LIST
VPN  IF NAME  ID     IP         VIRTUAL MAC        PRIORITY  STATE   STATE  TIMER          TIMER   LAST
STATE CHANGE TIME     LIST    STATE
-------------------------------------------------------------------------------------------------------------
```

```
-------------------------------------
1    ge0/6.2  2      10.2.2.1   00:0c:29:ab:b7:94  100       master  down   1              3       2015-
05-01T20:09:37+00:00  -       -
     ge0/6.3  3      10.2.3.11  00:00:00:00:00:00  100       init    down   1              3       0000-
00-00T00:00:00+00:00  -       -
```

In the following example, Router-1 is the VRRP primary, because it has a higher priority value than Router 2:

```
Router-1# show running-config vpn 1
vpn 1
!
 interface ge0/1.15
  ip address 10.10.1.2/24
  mtu         1496
  no shutdown
  vrrp 15
   priority  110
   track-omp
   ipv4 10.20.23.1
  !
 !
!

Router-1# show vrrp vpn 1
                                                                                                    MASTER
TRACK   PREFIX
              GROUP                                                 VRRP    OMP    ADVERTISEMENT  DOWN
PREFIX  LIST
VPN  IF NAME   ID     VIRTUAL IP     VIRTUAL MAC        PRIORITY  STATE   STATE  TIMER          TIMER
LAST STATE CHANGE TIME     LIST    STATE
-------------------------------------------------------------------------------------------------------------
-----------------------------------------
1    ge0/1.1   1      10.20.22.1     00:0c:bd:08:79:a4  100       backup  up     1              3
2016-01-13T03:10:55+00:00  -       -
     ge0/1.5   5      10.20.22.193   00:0c:bd:08:79:a4  100       backup  up     1              3
2016-01-13T03:10:55+00:00  -       -
     ge0/1.10  10     10.20.22.225   00:0c:bd:08:79:a4  100       backup  up     1              3
2016-01-13T03:10:55+00:00  -       -
     ge0/1.15  15     10.20.23.1     00:0c:bd:08:79:a4  110       master  up     1              3
2016-01-13T03:10:56+00:00  -       -
     ge0/1.20  20     10.20.24.1     00:0c:bd:08:79:a4  100       backup  up     1              3
2016-01-13T03:10:56+00:00  -       -
     ge0/1.25  25     10.20.25.1     00:0c:bd:08:79:a4  110       master  up     1              3
2016-01-13T03:10:56+00:00  -       -
     ge0/1.30  30     10.20.25.129   00:0c:bd:08:79:a4  100       backup  up     1              3
2016-01-13T03:10:56+00:00  -       -


Router-1# show vrrp vpn 1 interfaces ge0/1.15 groups 15

                                                                                        MASTER
TRACK   PREFIX
GROUP                                          VRRP    OMP    ADVERTISEMENT  DOWN
PREFIX  LIST
ID     VIRTUAL IP   VIRTUAL MAC        PRIORITY  STATE   STATE  TIMER          TIMER   LAST STATE CHANGE
TIME     LIST    STATE
-------------------------------------------------------------------------------------------------------------
--------------------------
1      10.20.33.1  00:0c:bd:08:79:a4  110       master  up     1              3       2016-01-
13T03:10:56+00:00  -       -

Router-2# show running-config vpn 1
vpn 1
!
```

```
 interface ge0/1.15
  ip address 10.10.1.3/24
  mtu         1496
  no shutdown
  vrrp 15
   track-omp
   ipv4 10.20.23.1
  !
 !
!

Router-2# show vrrp vpn 1 interfaces groups

                                                                                                                MASTER
TRACK   PREFIX
          GROUP                                                 VRRP    OMP    ADVERTISEMENT  DOWN
PREFIX  LIST
IF NAME   ID     VIRTUAL IP     VIRTUAL MAC        PRIORITY  STATE   STATE  TIMER          TIMER   LAST
STATE CHANGE TIME     LIST    STATE
-------------------------------------------------------------------------------------------------------------
-------------------------------------
ge0/1.1   1      10.20.32.1     00:0c:bd:08:2b:a5  110       master  up     1              3       2016-
01-13T00:22:15+00:00  -       -
ge0/1.5   5      10.20.32.193   00:0c:bd:08:2b:a5  110       master  up     1              3       2016-
01-13T00:22:15+00:00  -       -
ge0/1.10  10     10.20.32.225   00:0c:bd:08:2b:a5  110       master  up     1              3       2016-
01-13T00:22:15+00:00  -       -
ge0/1.15  15     10.20.33.1     00:0c:bd:08:2b:a5  100       backup  up     1              3       2016-
01-13T03:10:56+00:00  -       -
ge0/1.20  20     10.20.34.1     00:0c:bd:08:2b:a5  110       master  up     1              3       2016-
01-13T00:22:16+00:00  -       -
ge0/1.25  25     10.20.35.1     00:0c:bd:08:2b:a5  100       backup  up     1              3       2016-
01-13T03:10:56+00:00  -       -
ge0/1.30  30     10.20.35.129   00:0c:bd:08:2b:a5  100       master  up     1              3       2016-
01-13T00:22:16+00:00  -       -

Router-2# show vrrp vpn 100 interfaces groups 15

                                                                                                      MASTER
TRACK   PREFIX
          GROUP                                                 VRRP    OMP    ADVERTISEMENT  DOWN
PREFIX  LIST
IF NAME   ID     VIRTUAL IP     VIRTUAL MAC        PRIORITY  STATE   STATE  TIMER          TIMER   LAST
STATE CHANGE TIME     LIST    STATE
-------------------------------------------------------------------------------------------------------------
-------------------------------------
ge0/0.15  15     10.20.33.1     00:0c:bd:08:2b:a5  100       backup  up     1              3       2016-01-
13T03:10:56+00:00  -       -
```

Additional Information

Configuring Cellular Interfaces
Configuring DHCP
Configuring Network Interfaces
Configuring PPPoE
Configuring WLAN Interfaces

Configuring WLAN Interfaces

vEdge 100wm routers include a wireless LAN (WLAN) radio for providing access point (AP) functionality for teleworkers, small offices, stores, and branch offices. You can configure the radio to operate at either 2.4 GHz or 5 GHz. In 2.4-GHz mode, the radio can support IEEE 802.11b, 802.11g and 802.11n clients, and in 5-GHz mode, the radio can support IEEE 802.11a, 802.11n and 802.11ac clients. vEdge100wm

routers support 3x3 MIMO with three spatial streams, and they use an internal antenna. For WLAN security, you can use preshared key and RADIUS server–based methods.

This article describes how to configure the WLAN interfaces. To configure IEEE 802.11i authentication for the VAPs, see Configuring IEEE 802.1X and IEEE 802.11i Authentication .

Configure SSIDs

On a vEdge100wm router, you can configure up to four service set identifiers (SSIDs) on the WLAN radio. Each SSID is referred to by a virtual access point (VAP) interface. To a client, each VAP interface appears as a different access point (AP) with its own SSID. To provide access to different networks, you can assign each VAP to a different VLAN.

To configure a VAP interface that autoselects its channel and uses no authentication and no encyrption, create a VAP, assign it a number and an SSID, and enable it:

```
vEdge(config)# wlan radio-band
vEdge(config-wlan)# country country
vEdge(config-wlan)# interface vapnumber
vEdge(config-vap)# no shutdown
vEdge(config-vap)# ssid ssid
```

For the radio band, specify one of the following:

- **2.4GHz** —Consists of fourteen 20-MHz channels with overlapping frequency space. The allowable channels and maximum allowed output power are country specific and restricted by regulatory agencies. In the United States and Canada, channels 1, 6 ,and 11 are the only non-overlapping channels. This radio band supports IEEE 802.11b, 802.11g, and 802.11n clients..
- **5GHz** —Consists of four 20-MHz channels in UNII-1, four in UNII-2, twelve in UNII-2 Extended, four in UNII-3 and one in ISM band. The allowable channels, their indoor or outdoor usage, and the maximum allowed output power are country specific and are restricted by regulatory agencies. This radio band supports IEEE 802.11a, 802.11n, and 802.11ac clients.

Configuring the country where the router is installed is mandatory, to ensure that the router complies to local regulatory requirements.

For each SSID, configure one VAP interface. *number* can be from 0 through 3. To reduce RF congestion, it is recommended that you do not configure more than two VAP interfaces on the router.

To activate (enable) the VAP interface, include the **no shutdown** command.

Each VAP has an SSID. For *ssid* , enter the name of the SSID. It can be a string from 4 through 32 characters. The SSID must be unique.

By default, a maximum of 25 clients can connect to a single VAP. You can change the maximum number of clients to a value from 1 through 50:

```
vEdge(config-vap)# max-clients number
```

It is recommended that you do not configure more than 50 clients across all the VAPs.

Configure Radio-Specific Parameters

For each radio band, you can configure radio-specific parameters.

Specify the country where the router is installed. This configuration is mandatory and ensures that the router complies to local regulatory requirements, and it enforces country-specific allowable channels and maximum allowed output power. By default, the country is the United States. To set a different country, specify the country where the router is installed:

```
vEdge(config-wlan)# country country
```

You can use the Viptela wireless router software for the following countries: Australia, Austria, Belgium, Brazil, Bulgaria, Canada, China, Costa Rica, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hong Kong, Hungary, Iceland, India,

Indonesia, Ireland, Italy, Japan, Latvia, Liechtenstein, Lithuania, Luxembourg, Malaysia, Malta, Mexico, Netherlands, New Zealand, Norway, Pakistan, Panama, Philippines, Poland, Portugal, Puerto Rico, Romania, Saudi Arabia, Singapore, Slovakia, Slovenia, South Africa, South Korea, Spain, Sri Lanka, Sweden, Switzerland, Taiwan, Thailand, Turkey, United Kingdom, United States, and Vietnam.

Note: Check the release notes for your software release to determine the countries in which the vEdge 100wm router is certified.

By default, the best radio channel is selected automatically. To explicitly configure automatic channel selection, use the following command:

```
vEdge(config-wlan)# channel auto
```

To configure automatic channel selection that excludes channels with dynamic frequency selection (DFS) capabilities, use the following command:

```
vEdge(config-wlan)# channel auto-no-dfs
```

To explicitly configure the radio channel to use:

```
vEdge(config-wlan)# channel channel
```

For 2.4-GHz WLANs, the channel can be 1 through 13, depending on the country configuration.

For 5-GHz WLANs, the channel, including DFS channels, can be one of 36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144, 149, 153, 157, 161, and 165, depending on the country configuration.

Note: Airport radar uses frequencies that overlap DFS channels. If you are using a 5-GHz radio band, and if your installation is near an airport, it is recommended that you configure **channel auto-no-dfs** , to remove DFS channels from the list of available channels.

By default, 2.4-GHz radio bands are allocated a channel bandwidth of 20 MHz, and 5-GHz radio bands have a channel bandwidth of 80 MHz. You can set the bandwidth to 20, 40, or 80 MHz:

```
vEdge(config-wlan)# channel-bandwidth megahertz
```

The guard interval is the time between symbol transmissions on the WLAN. For 2.4-GHz radio frequencies, the default guard interval is 800 nanoseconds (which is the normal guard interval), and for 5-GHz frequencies it is 400 nanoseconds (which is the short guard interval). These are the only two guard intervals available. The short guard interval can increase throughput, but it can also increase the error rate because of increased sensitivity to RF reflections. You can choose to configure the guard interval explicitly:

```
vEdge(config-wlan)# guard-interval nanoseconds
```

Configure a Bridging Domain and IRB

To provide the SSIDs access to different networks, you can assign each VAP to a different VLAN. To do this, create one bridge domain with an untagged VLAN for each SSID:

```
vEdge(config)# bridge number
vEdge(config-bridge)# interface vapnumber
vEdge(config-vap)# no native-vlan
vEdge(config-vap)# no shutdown
```

To allow data traffic to be passed among different VLANs, you create an integrated routing and bridging (IRB) logical interface in a VPN domain that connects to the bridging domain:

```
vEdge(config)# vpn vpn-id
vEdge(config-vpn)# interface irbnumber
vEdge(config-irb)# ip address prefix/length
vEdge(config-irb)# no shutdown
```

Configure a DHCP server on the IRB interface so that clients connecting to the VLAN can receive IP addresses in the VLAN:

```
vEdge(config-irb)# dhcp-server
vEdge(config-dhcp-server)# address-pool prefix/length
vEdge(config-dhcp-server)# admin-state (down | up)
```

```
vEdge(config-dhcp-server)# options
vEdge(config-options)# default-gateway ip-address
```

WLAN Interface Configuration Example

The configuration example in this section shows how to configure two SSIDs on a WLAN router. One SSID is called CorporateNetwork, and the second is called GuestNetwork.

First, configure the WLAN radio band, and within it, create two VAP interfaces, one for each SSID:

```
wlan 5GHz
 country "United States"
 interface vap0
  ssid             CorporateNetwork
  data-security    wpa/wpa2-enterprise
  radius-server radius_server1
  max-clients      30
  no shutdown
 !
 interface vap1
  ssid             GuestNetwork
  data-security    wpa/wpa2-personal
  wpa-personal-key GuestPassword
  max-clients      10
  no shutdown
 !
!
```

The CorporateNetwork SSID uses wpa/wpa2-enterprise data security, which works in conjunction with a RADIUS authentication server. Here is the configuration for the RADIUS server:

```
system
 radius
  server 10.20.24.15
   acct-port  0
   tag        radius_server1
   vpn        1
   secret-key radiusSecretKey
  exit
 !
!
```

Next, configuring two bridging domains, one for each VAP interface (that is, one for each SSID):

```
bridge 1
 interface vap0
  no native-vlan
  no shutdown
 !
!
bridge 2
 interface vap1
  no native-vlan
  no shutdown
 !
!
```

Finally, configure IRB interfaces and the DHCP server. Here, the SSID CorporateNetwork uses VPN 1, and the GuestNetwork uses VPN 100:

```
vpn 1
 name "Corporate Network"
 interface irb1
  ip address 10.30.30.1/24
```

```
    no shutdown
    dhcp-server
     address-pool 10.30.30.0/24
     offer-time   600
     lease-time   86400
     admin-state  up
     options
      default-gateway 10.30.30.1
      dns-servers     8.8.8.8
     !
    !
   !
  !
  vpn 100
   name "Guest Network"
   interface irb2
    ip address 192.168.30.1/24
    no shutdown
    dhcp-server
     address-pool 192.168.30.0/24
     offer-time   600
     lease-time   86400
     admin-state  up
     options
      default-gateway 192.168.30.1
      dns-servers     8.8.8.8
     !
    !
   !
   ip route 0.0.0.0/0 vpn 0
  !
```

Additional Information

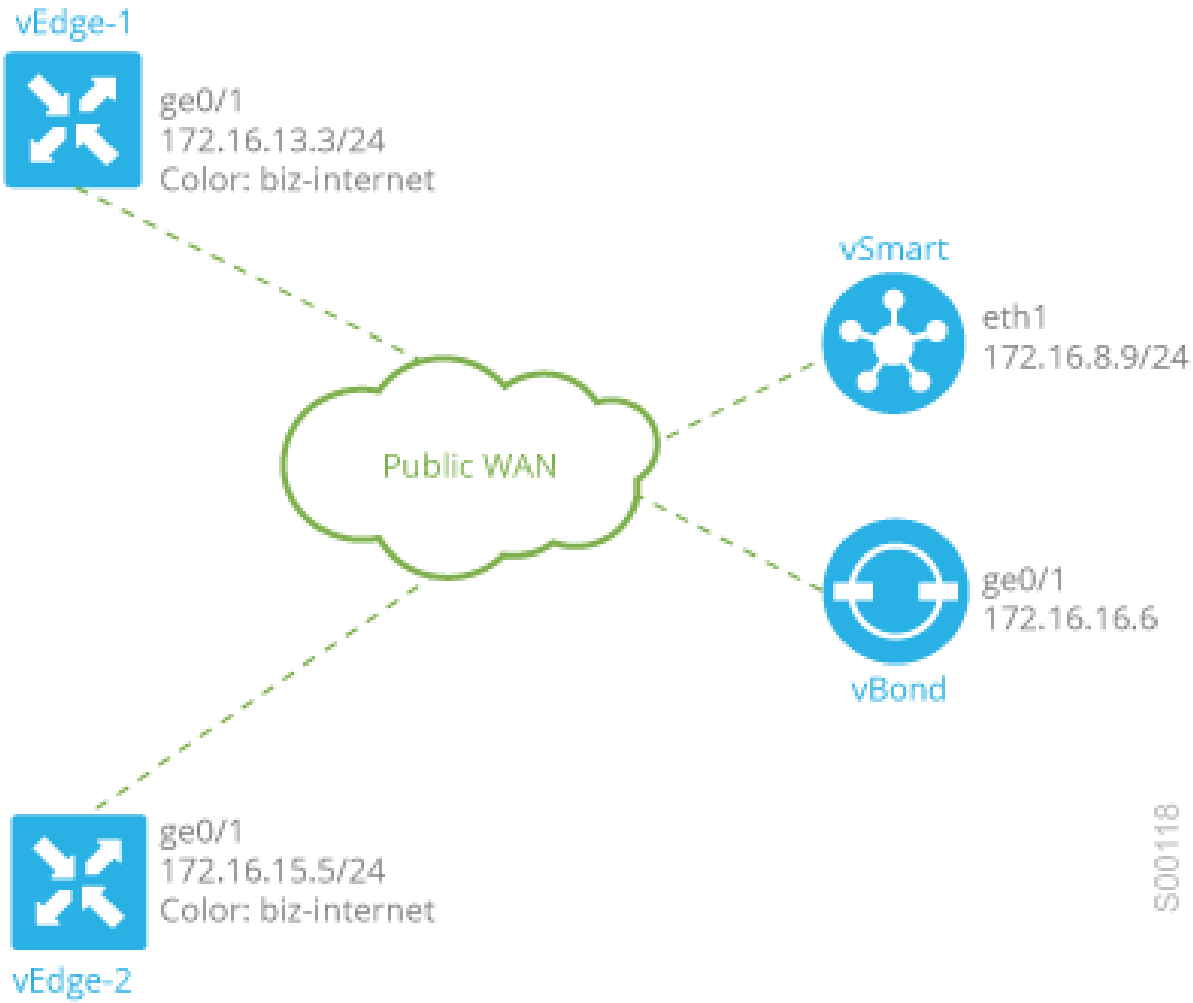
Configuring Cellular Interfaces
Configuring DHCP
Configuring IEEE 802.1X and IEEE 802.11i Authentication
Configuring Network Interfaces
Configuring PPPoE
Configuring VRRP

Network Interface Configuration Examples

This article provides examples of configuring interfaces on vEdge routers to allow the flow of data traffic across both public and private WAN transport networks.

Connect to a Public WAN

This example shows a basic configuration for two vEdge routers connected to the same public WAN network (such as the Internet). The vSmart controller and vBond orchestrator are also connected to the public WAN network, and the vSmart controller is able to reach all destinations on the public WAN.



For vEdge-1, the interface ge0/1 connects to the public WAN, so it is the interface that is configured as a tunnel interface. The tunnel has a color of biz-internet, and the encapsulation used for data traffic is IPsec. The Viptela software creates a single TLOC for this interface, comprising the interface's IP address, color, and encapsulation, and the TLOC is sent to the vSmart controller over the OMP session running on the tunnel. The configuration also includes a default route to ensure that the router can reach the vBond orchestrator and vSmart controller.

```
vpn 0
  interface ge0/1
    ip address 172.16.13.3/24
    tunnel-interface
      encapsulation ipsec
      color biz-internet
      allow-service dhcp
      allow-service dns
      allow-service icmp
      no allow-service sshd
      no allow-service ntp
      no allow-service stun
    !
    no shutdown
  !
```

```
  ip route 0.0.0.0/0 172.16.13.1
!
```

The configuration for vEdge-2 is similar to that for vEdge-1:

```
vpn 0
  interface ge0/1
    ip address 172.16.15.5/24
    tunnel-interface
      encapsulation ipsec
      color biz-internet
      allow-service dhcp
      allow-service dns
      allow-service icmp
      no allow-service sshd
      no allow-service ntp
      no allow-service stun
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.15.1
!
```

On the vSmart controller and vBond orchestrator, you configure a tunnel interface and default IP route to reach the WAN transport. For the tunnel, color has no meaning because these devices have no TLOCs.

```
vpn 0
  interface eth1
    ip address 172.16.8.9/24
      tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.8.1
!
vpn 0
  interface ge0/1
    ip address 172.16.16.6/24
      tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.16.1
!
```

Use the **show interface** command to check that the interfaces are operational and that the tunnel connections have been established. In the Port Type column, tunnel connections are marked as "transport."

```
vEdge-1# show interface vpn 0

                                    IF      IF
TCP
                                    ADMIN   OPER    ENCAP                                       SPEED
MSS                  RX       TX
VPN  INTERFACE  IP ADDRESS          STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR             MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------
0    ge0/0      172.16.13.3/24      Up      Up      null   transport  1500  00:0c:29:7d:1e:fe  10     full
0       0:02:26:20  88358    88202
0    ge0/1      10.1.17.15/24       Up      Up      null   service    1500  00:0c:29:7d:1e:08  10     full
0       0:02:26:20  217      1
0    ge0/2      -                   Down    Up      null   service    1500  00:0c:29:7d:1e:12  10     full
0       0:02:26:20  217      0
0    ge0/3      10.0.20.15/24       Up      Up      null   service    1500  00:0c:29:7d:1e:1c  10     full
0       0:02:26:20  218      1
```

```
0    ge0/6      57.0.1.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:3a  10     full
0       0:02:26:20  217      1
0    ge0/7      10.0.100.15/24    Up      Up      null   service    1500  00:0c:29:7d:1e:44  10     full
0       0:02:25:02  850      550
0    system     172.16.255.3/32   Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:02:13:31  0        0
```

Use the **show control connections** command to check that the vEdge router has a DTLS or TLS session established to the vSmart controller.

```
vEdge-1# show control connections
                                                                                    PEER
PEER
PEER     PEER     PEER             SITE        DOMAIN      PEER             PRIVATE  PEER
PUBLIC
TYPE     PROTOCOL SYSTEM IP        ID          ID          PRIVATE IP       PORT     PUBLIC IP
PORT    LOCAL COLOR      STATE           UPTIME
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------
vsmart   dtls     172.16.255.19    100         1           10.0.5.19        12346    10.0.5.19
12346   biz-internet     up                  0:02:13:13
vsmart   dtls     172.16.255.20    200         1           10.0.12.20       12346    10.0.12.20
12346   biz-internet     up                  0:02:13:13
```

Use the **show bfd sessions** command to display information about the BFD sessions that have been established between the local vEdge router and remote routers:

```
vEdge-1# show bfd sessions
                                       SOURCE TLOC      REMOTE TLOC                        DST PUBLIC
DST PUBLIC         DETECT      TX
SYSTEM IP        SITE ID  STATE       COLOR            COLOR            SOURCE IP        IP
PORT        ENCAP  MULTIPLIER  INTERVAL(msec)  UPTIME         TRANSITIONS
-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------
172.16.255.11    100      up          biz-internet     biz-internet     10.1.15.15       10.0.5.11
12346       ipsec  20          1000          0:02:24:59      1
172.16.255.14    400      up          biz-internet     biz-internet     10.1.15.15       10.1.14.14
12360       ipsec  20          1000          0:02:24:59      1
172.16.255.16    600      up          biz-internet     biz-internet     10.1.15.15       10.1.16.16
12346       ipsec  20          1000          0:02:24:59      1
172.16.255.21    100      up          biz-internet     biz-internet     10.1.15.15       10.0.5.21
12346       ipsec  20          1000          0:02:24:59      1
```

Use the **show omp tlocs** command to list the TLOCs that the local router has learned from the vSmart controller:

```
vEdge-1# show omp tlocs
C   -> chosen
I   -> installed
Red -> redistributed
Rej -> rejected
L   -> looped
R   -> resolved
S   -> stale
Ext -> extranet
Inv -> invalid

ADDRESS                                                                                    PUBLIC
PRIVATE  BFD
FAMILY   TLOC IP          COLOR            ENCAP  FROM PEER        STATUS    PUBLIC IP        PORT
PRIVATE IP       PORT     STATUS
-------------------------------------------------------------------------------------------------------------
-------------------------------
ipv4     172.16.255.11    biz-internet     ipsec  172.16.255.19    C,I,R     10.0.5.11        12346
10.0.5.11        12346    up
                                                  172.16.255.20    C,R       10.0.5.11        12346
```

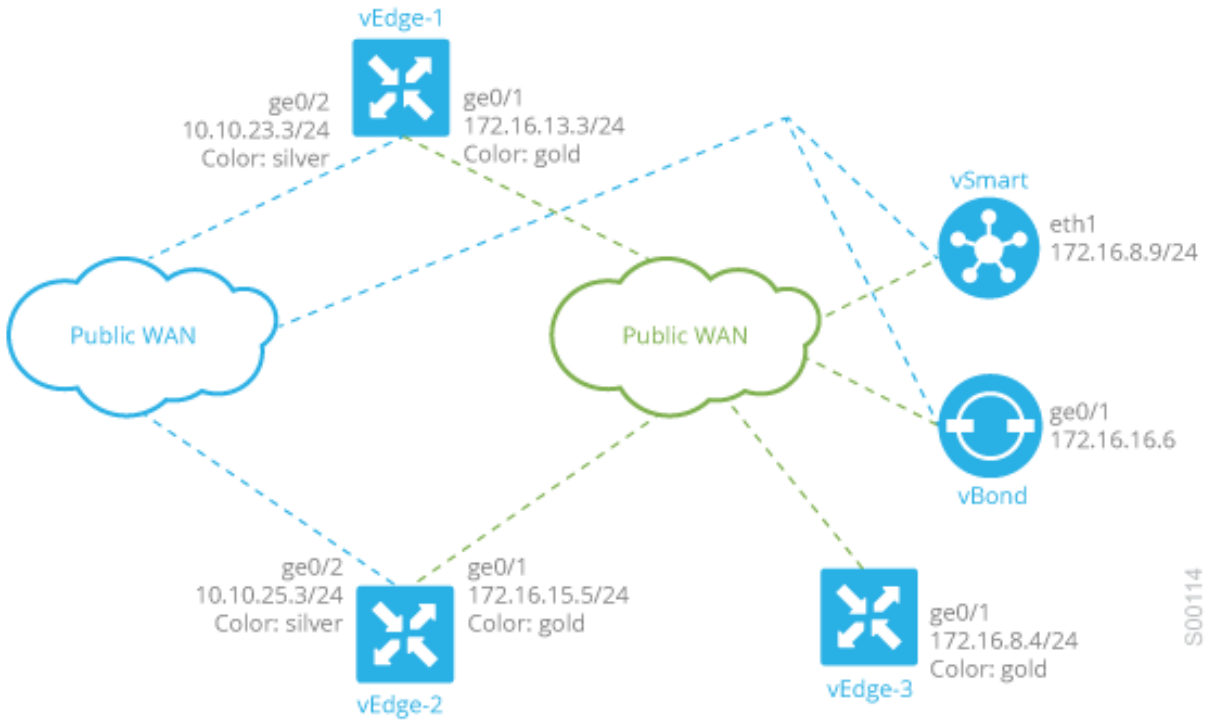
```
10.0.5.11         12346    up
         172.16.255.14    biz-internet     ipsec  172.16.255.19    C,I,R     10.1.14.14        12360
10.1.14.14        12360    up
                                                   172.16.255.20    C,R       10.1.14.14        12360
10.1.14.14        12360    up
         172.16.255.16    biz-internet     ipsec  172.16.255.19    C,I,R     10.1.16.16        12346
10.1.16.16        12346    up
                                                   172.16.255.20    C,R       10.1.16.16        12346
10.1.16.16        12346    up
         172.16.255.21    biz-internet     ipsec  172.16.255.19    C,I,R     10.0.5.21         12346
10.0.5.21         12346    up
                                                   172.16.255.20    C,R       10.0.5.21         12346
10.0.5.21         12346    up    <
```

Connect to Two Public WANs

In this example, two vEdge routers at two different sites connect to two public WANs, and hence each router has two tunnel connections. To direct traffic to the two different WANs, each tunnel interface is assigned a different color (here, **silver** and **gold**). Because each router has two tunnels, each router has two TLOCs.

A third router at a third site, vEdge-3, connects only to one of the public WANs.

The vSmart controller and vBond orchestrator are connected to one of the public WAN networks. (In reality, it does not matter which of the two networks they are connected to, nor does it matter whether the two devices are connected to the same network.) The vSmart controller is able to reach all destinations on the public WAN. To ensure that the vBond orchestrator is accessible via each transport tunnel on the routers, a default route is configured for each interface. In our example, we configure a static default route, but you can also use DHCP.



The configurations for vEdge-1 and vEdge-2 are similar. We configure two tunnel interfaces, one with color **silver** and the other with color **gold** , and we configure static default routes for both tunnel interfaces. Here is the configuration for vEdge-1:

```
vpn 0
  interface ge0/1
```

```
    ip address 172.16.13.3/24
    tunnel-interface
      encapsulation ipsec
      color silver
    !
    no shutdown
  !
  interface ge0/2
    ip address 10.10.23.3/24
    tunnel-interface
      encapsulation ipsec
      color gold
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.13.1
  ip route 0.0.0.0/0 10.10.23.1
```

The configuration for vEdge-2 is similar:

```
vpn 0
  interface ge0/1
    ip address 172.16.15.5/24
    tunnel-interface
      encapsulation ipsec
      color silver
    !
    no shutdown
  !
  interface ge0/2
    ip address 10.10.25.3/24
    tunnel-interface
      encapsulation ipsec
      color gold
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.15.1
  ip route 0.0.0.0/0 10.10.25.1
```

The third router, vEdge-3, connects only to one of the public WAN networks, and its tunnel interface is assigned the color "gold":

```
vpn 0
  interface ge0/1
    ip address 172.16.8.4/24
    tunnel-interface
      encapsulation ipsec
      color gold
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.8.1
```

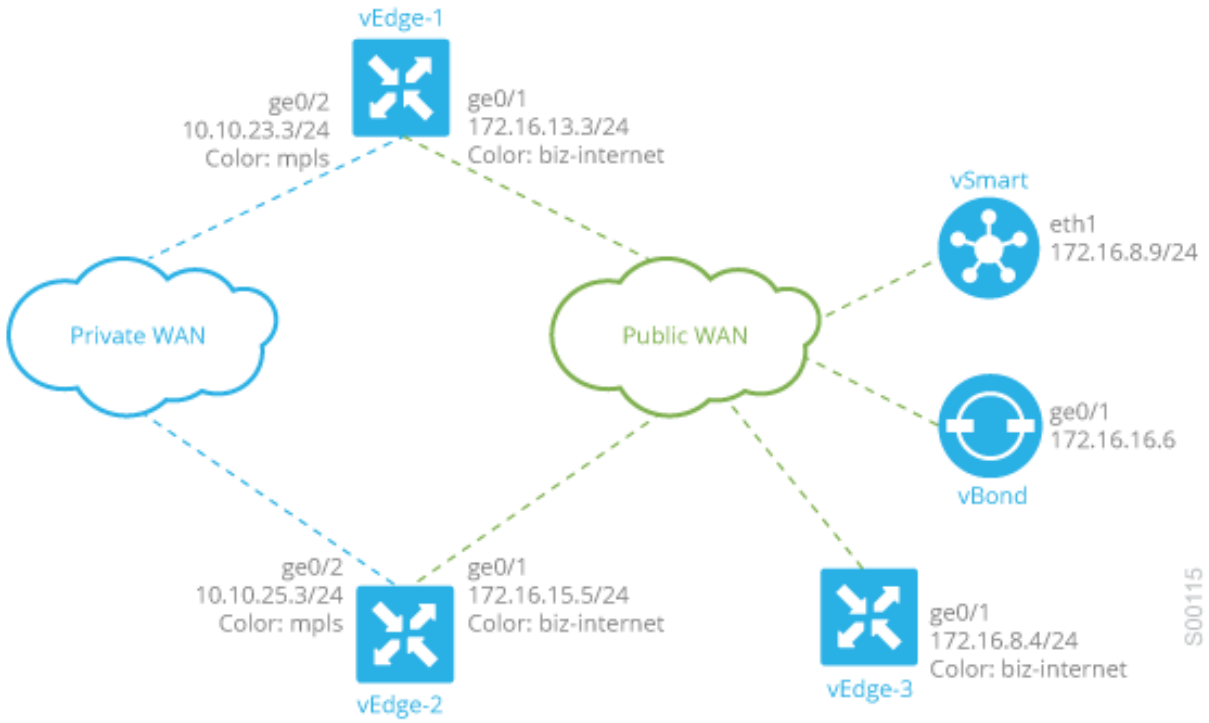
On the vSmart controller and vBond orchestrator, you configure a tunnel interface and default IP route to reach the WAN transport. For the tunnel, color has no meaning because these devices have no TLOCs.

```
vpn 0
  interface eth1
    ip address 172.16.8.9/24
      tunnel-interface
    !
    no shutdown
  ip route 0.0.0.0/0 172.16.8.1
vpn 0
  interface ge0/1
```

```
    ip address 172.16.16.6/24
      tunnel-interface
    !
    no shutdown
ip route 0.0.0.0/0 172.16.16.1
```

Connect to Public and Private WANs, with Separation of Network Traffic

In this example, two vEdge routers at two different sites each connect to the same public WAN (here, the Internet) and the same private WAN (here, an MPLS network). We want to separate the MPLS network completely so that it is not reachable by the Internet. The vSmart controller and vBond orchestrator are hosted in the provider's cloud, which is reachable only via the Internet. A third vEdge router at a third site connects only to the public WAN (Internet).



In this example topology, we need to ensure the following:

- Complete traffic separation exists between private-WAN (MPLS) traffic and public-WAN (Internet) traffic.
- Each site (that is, each vEdge router) must have a connection to the Internet, because this is the only way that the overlay network can come up.

To maintain complete separation between the public and private networks so that all MPLS traffic stays within the MPLS network, and so that only public traffic passes over the Internet, we create two overlays, one for the private MPLS WAN and the second for the public Internet. For the private overlay, we want to create data traffic tunnels (which run IPsec and BFD sessions) between private-WAN TLOCs, and for the public overlay we want to create these tunnel connections between Internet TLOCs. To make sure that no data traffic tunnels are established between private-WAN TLOCs and Internet TLOCs, or vice versa, we associate the **restrict** attribute with the color on the private-WAN TLOCs. When a TLOC is marked as restricted, a TLOC on the local router establishes tunnel connections with a remote TLOC only if the remote TLOC has the same color. Put another way, BFD sessions come up between two private-WAN TLOCs and they come up between two public-WAN TLOCs, but they do not come up between an MPLS TLOC and an Internet TLOC.

Each site must have a connection to the public (Internet) WAN so that the overlay network can come up. In this topology, the vSmart controller and vBond orchestrator are reachable only via the Internet, but the MPLS network is completely isolated from the Internet. This means that if a vEdge router were to connect just to the MPLS network, it would never be able to discover the vSmart and vBond devices

and so would never be able to never establish control connections in the overlay network. In order for a vEdge router in the MPLS network to participate in overlay routing, it must have at least one tunnel connection, or more specifically, one TLOC, to the Internet WAN. (Up to seven TLOCs can be configured on each vEdge router.) The overlay network routes that the router router learns over the public-WAN tunnel connection populate the routing table on the vEdge router and allow the router and all its interfaces and TLOCs to participate in the overlay network.

By default, all tunnel connections attempt to establish control connections in the overlay network. Because the MPLS tunnel connections are never going to be able to establish these connections to the vSmart or vBond devices, we include the **max-control-connections 0** command in the configuration. While there is no harm in having the MPLS tunnels attempt to establish control connections, these attempts will never succeed, so disabling them saves resources on the vEdge router. Note that **max-control-connections 0** command works only when there is no NAT device between the vEdge router and the PE router in the private WAN.

Connectivity to sites in the private MPLS WAN is possible only by enabling service-side routing.

Here is the configuration for the tunnel interfaces on vEdge-1. This snippet does not include the service-side routing configuration.

```
vpn 0
  interface ge0/1
    ip address 172.16.13.3/24
    tunnel-interface
      encapsulation ipsec
      color biz-internet
    !
    no shutdown
  !
  interface ge0/2
    ip address 10.10.23.3/24
    tunnel-interface
      encapsulation ipsec
      color mpls restrict
      max-control-connections 0
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.13.1
```

The configuration on vEdge-2 is quite similar:

```
vpn 0
  interface ge0/1
    ip address 172.16.15.5/24
    tunnel-interface
      encapsulation ipsec
      color biz-internet
    !
    no shutdown
  !
  interface ge0/2
    ip address 10.10.25.3/24
    tunnel-interface
      encapsulation ipsec
      color mpls restrict
      max-control-connections 0
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.15.1
!
```

The vEdge-3 router connects only to the public Internet WAN:

```
vpn 0
  interface ge0/1
    ip address 172.16.8.4/24
```

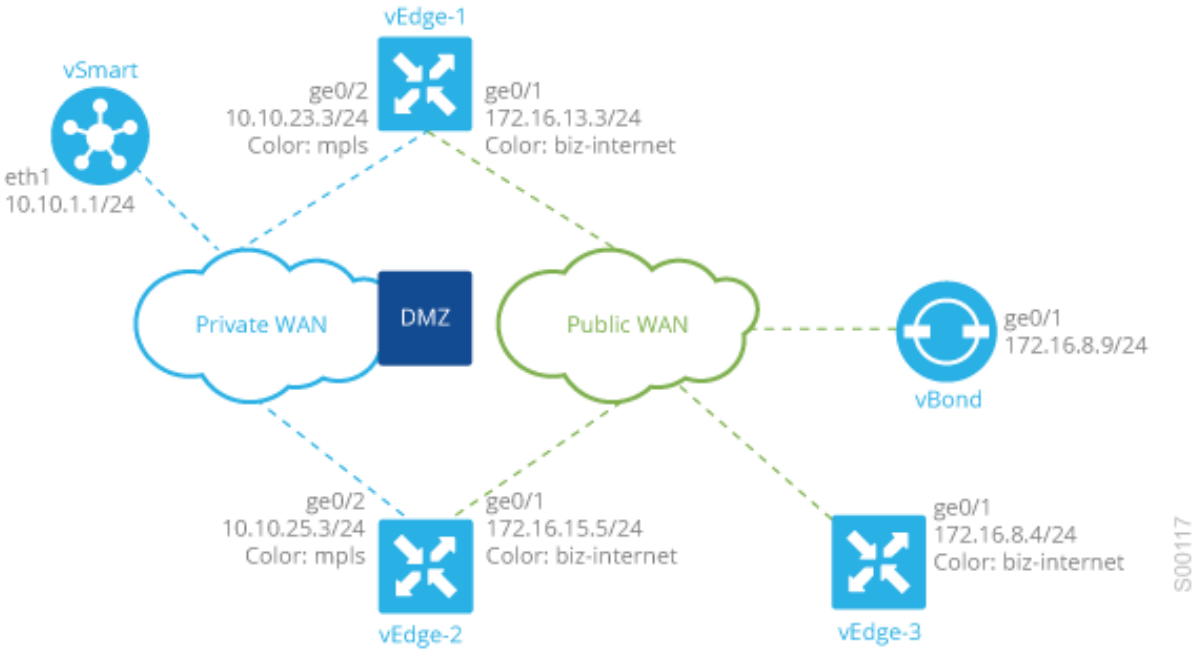
```
    tunnel-interface
      encapsulation ipsec
      color biz-internet
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.8.1
!
```

On the vSmart controller and vBond orchestrator, you configure a tunnel interface and default IP route to reach the WAN transport. For the tunnel, color has no meaning because these devices have no TLOCs.

```
vpn 0
  interface eth1
    ip address 172.16.8.9/24
      tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.8.1
!
vpn 0
  interface ge0/1
    ip address 172.16.16.6/24
      tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.16.1
!
```

Connect to Public and Private WANs, with Ubiquitous Connectivity to Both WANs

This example is a variant of the previous example. We still have two vEdge routers at two different sites each connect to the same public WAN (here, the Internet) and the same private WAN (here, an MPLS network). However, now we want sites on the MPLS network and the Internet to be able to exchange data traffic. This topology requires a single overlay over both the public and private WANs. Control connections are present over both transports, and we want IPsec tunnel connections running BFD sessions to exist from private-WAN TLOCs to private-WAN TLOCs, from Internet TLOCs to Internet TLOCs, from private-WAN TLOCs to Internet TLOCs, and from Internet TLOCs to private-WAN TLOCs. This full possibility of TLOCs allows the establishment of a ubiquitous data plane in the overlay network.



For this configuration to work, the vBond orchestrator must be reachable over both WAN transports. Because it is on the public WAN (that is, on the Internet), there needs to be connectivity from the private WAN to the Internet. This could be provided via a DMZ, as shown in the figure above. The vSmart controller can be either on the public or the private LAN. If there are multiple controllers, some can be on public LAN and others on private LAN.

On each vEdge router, you configure private-WAN TLOCs, assigning a private color (**metro-ethernet** , **mpls** , or **private1** through **private6**) to the tunnel interface. You also configure public TLOCs, assigning any other color (or you can leave the color as **default**). Each vEdge router needs two routes to reach the vBond orchestrator, one via the private WAN and one via the public WAN.

With such a configuration:

- Control connections are established over each WAN transport.
- BFD/IPsec comes up between all TLOCs (if no policy is configured to change this).
- A given site can be dual-homed to both WAN transports or single-homed to either one.

Here is an example of the configuration on one of the vEdge routers, vEdge-1:

```
vpn 0
  interface ge0/1
    description "Connection to public WAN"
    ip address 172.16.31.3/24
    tunnel-interface
      encapsulation ipsec
      color biz-internet
    !
    no shutdown
  !
  interface ge0/2
    description "Connection to private WAN"
    ip address 10.10.23.3/24
    tunnel-interface
      encapsulation ipsec
      color mpls
    !
    no shutdown
```

```
  !
  ip route 0.0.0.0/0 10.10.23.1
  ip route 0.0.0.0/0 172.16.13.1
!
```

The **show control connections** command lists two DTLS sessions to the vSmart controller, one from the public tunnel (color of **biz-internet**) and one from the private tunnel (color of **mpls**):

```
vEdge-1# show control connections
                                                                                    PEER
PEER
PEER     PEER     PEER             SITE        DOMAIN      PEER             PRIVATE  PEER
PUBLIC
TYPE     PROTOCOL SYSTEM IP        ID          ID          PRIVATE IP       PORT     PUBLIC IP
PORT    LOCAL COLOR      STATE           UPTIME
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------
vsmart   dtls     1.1.1.9          900         1           172.16.8.2       12346    172.16.8.2
12346   mpls             up                  0:01:41:17
vsmart   dtls     1.1.1.9          900         1           172.16.8.2       12346    172.16.8.2
12346   biz-internet     up                  0:01:41:33
```

The **show bfd sessions** command output shows that vEdge-1 has separate tunnel connections that are running separate BFD sessions for each color:

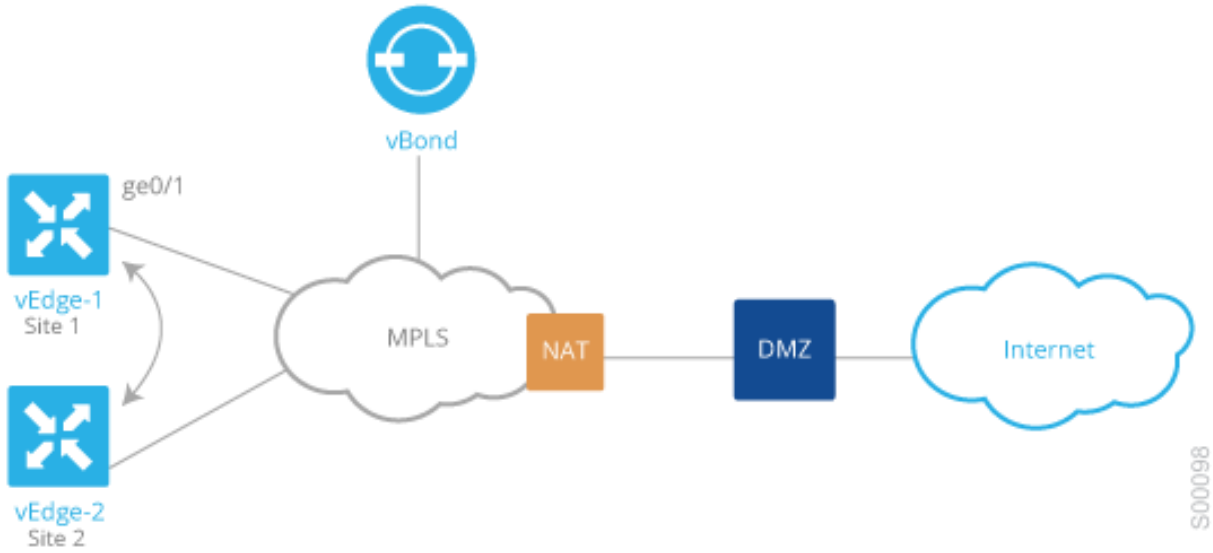
```
vEdge-1# show bfd sessions
                                       SOURCE TLOC      REMOTE TLOC                       DST PUBLIC
DST PUBLIC          DETECT      TX
SYSTEM IP        SITE ID  STATE       COLOR            COLOR            SOURCE IP        IP
PORT        ENCAP  MULTIPLIER  INTERVAL(msec)  UPTIME         TRANSITIONS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------
1.1.1.5          500      up          mpls             biz-internet     10.10.23.3       172.16.51.5
12346       ipsec  3           1000          0:06:07:19      1
1.1.1.5          500      up          biz-internet     biz-internet     172.16.31.3      172.16.51.5
12360       ipsec  3           1000          0:06:07:19      1
1.1.1.6          600      up          mpls             biz-internet     10.10.23.3       172.16.16.6
12346       ipsec  3           1000          0:06:07:19      1
1.1.1.6          600      up          biz-internet     biz-internet     172.16.31.3      172.16.16.6
12346       ipsec  3           1000          0:06:07:19      1
```

Exchange Data Traffic within a Single Private WAN

When the vEdge router is connected is a private WAN network, such as an MPLS or a metro Ethernet network, and when the carrier hosting the private network does not advertise the router's IP address, remote vEdge routers on the same private network but at different sites can never learn how to reach that router and hence are not able to exchange data traffic with it by going only through the private network. Instead, the remote routers must route data traffic through a local NAT and over the Internet to a vBond orchestrator, which then provides routing information to direct the traffic to its destination. This process can add significant overhead to data traffic exchange, because the vBond orchestrator may physically be located at a different site or a long distance from the two vEdge routers and because it may be situated behind a DMZ.

To allow vEdge routers at different overlay network sites on the private network to exchange data traffic directly using their private IP addresses, you configure their WAN interfaces to have one of eight private colors, **metro-ethernet** , **mpls** , and **private1** through **private6** . Of these four colors, the WAN interfaces on the vEdge routers must be marked with the same color so that they can exchange data traffic.

To illustrate the exchange of data traffic across private WANs, let's look at a simple topology in which two vEdge routers are both connected to the same private WAN. The following figure shows that these two vEdge routers are connected to the same private MPLS network. The vEdge-1 router is located at Site 1, and vEdge-2 is at Site 2. Both routers are directly connected to PE routers in the carrier's MPLS cloud, and you want both routers to be able to communicate using their private IP addresses.



This topology requires a special configuration to allow traffic exchange using private IP addresses because:

- The vEdge routers are in different sites; that is, they are configured with different site IDs.
- The vEdge routers are directly connected to the PE routers in the carrier's MPLS cloud.
- The MPLS carrier does not advertise the link between the vEdge router and its PE router.

To be clear, if the situation were one of the following, no special configuration would be required:

- vEdge-1 and vEdge-2 are configured with the same site ID.
- vEdge-1 and vEdge-2 are in different sites, and the vEdge router connects to a CE router that, in turn, connects to the MPLS cloud.
- vEdge-1 and vEdge-2 are in different sites, the vEdge router connects to the PE router in the MPLS cloud, and the private network carrier advertises the link between the vEdge router and the PE router in the MPLS cloud.
- vEdge-1 and vEdge-2 are in different sites, and you want them to communicate using their public IP addresses.

In this topology, because the MPLS carrier does not advertise the link between the vEdge router and the PE router, you use a loopback interface on the each vEdge router to handle the data traffic instead of using the physical interface that connects to the WAN. Even though the loopback interface is a virtual interface, when you configure it on the vEdge router, it is treated like a physical interface: the loopback interface is a terminus for both a DTLS tunnel connection and an IPsec tunnel connection, and a TLOC is created for it.

This loopback interface acts as a transport interface, so you must configure it in VPN 0.

For the vEdge-1 and vEdge-2 routers to be able to communicate using their private IP addresses over the MPLS cloud, you set the color of their loopback interfaces to be the same and to one of eight special colors— **metro-ethernet** , **mpls** , and **private1** through **private6** .

Here is the configuration on vEdge-1:

```
vedge-1(config)# vpn 0
vedge-1(config-vpn-0)# interface loopback1
vedge-1(config-interface-loopback1)# ip address 172.16.255.25/32
vedge-1(config-interface-loopback1)# tunnel-interface
vedge-1(config-tunnel-interface)# color mpls
vedge-1(config-interface-tunnel-interface)# exit
vedge-1(config-tunnel-interface)# no shutdown
vedge-1(config-tunnel-interface)# commit and-quit
vedge-1# show running-config vpn 0
```

```
...
 interface loopback1
  ip-address 172.16.255.25/32
  tunnel-interface
   color mpls
  !
  no shutdown
 !
```

On vEdge-2, you configure a loopback interface with the same tunnel interface color that you used for vEdge-1:

```
vedge-2# show running-config vpn 0
vpn 0
 interface loopback2
  ip address 172. 17.255.26/32
  tunnel-interface
   color mpls
  no shutdown
 !
```

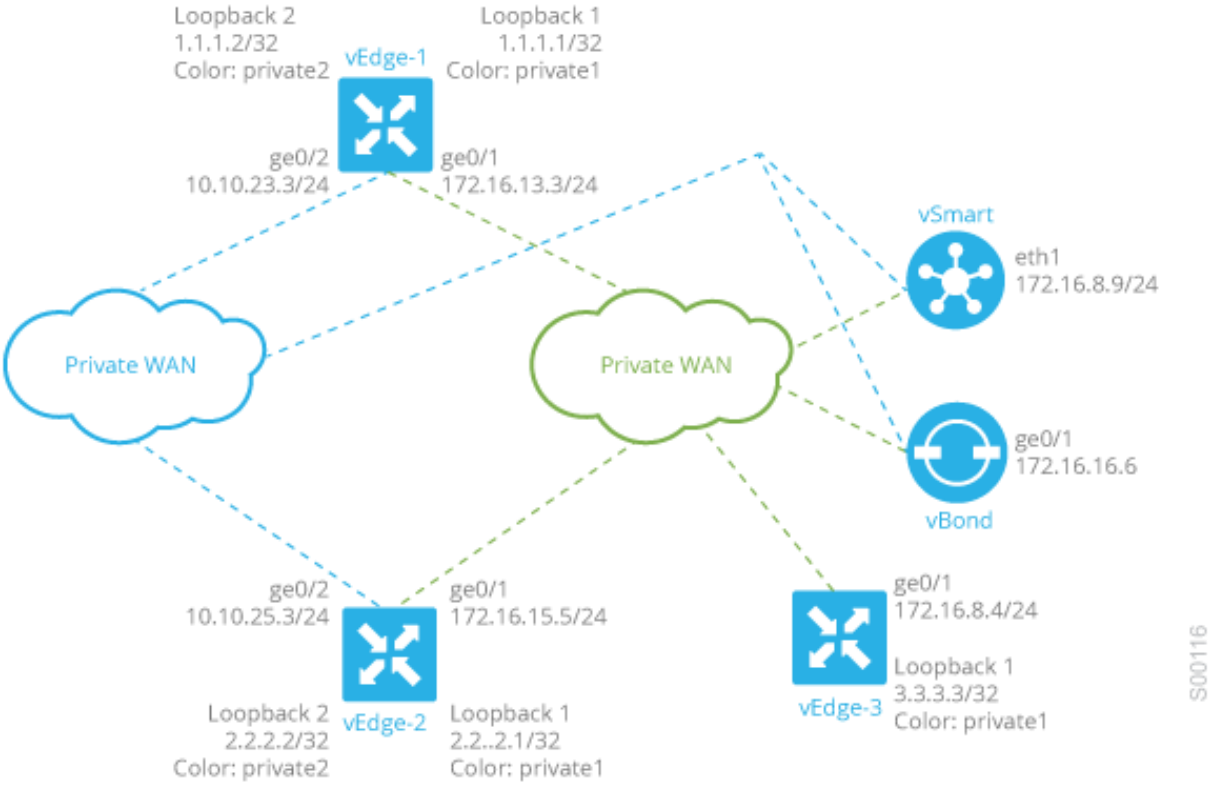
Use the **show interface** command to verify that the loopback interface is up and running. The output shows that the loopback interface is operating as a transport interface, so this is how you know that it is sending and receiving data traffic over the private network.

```
vedge-1# show interface
                                    IF      IF
TCP
                                    ADMIN   OPER    ENCAP                                       SPEED
MSS                  RX       TX
VPN  INTERFACE  IP ADDRESS          STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR             MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------
0    ge0/0      10.1.15.15/24       Up      Up      null   transport  1500  00:0c:29:7d:1e:fe  10     full
0       0:07:38:49  213199   243908
0    ge0/1      10.1.17.15/24       Up      Up      null   service    1500  00:0c:29:7d:1e:08  10     full
0       0:07:38:49  197      3
0    ge0/2      -                   Down    Down    null   service    1500  00:0c:29:7d:1e:12  -      -
0       -          1        1
0    ge0/3      10.0.20.15/24       Up      Up      null   service    1500  00:0c:29:7d:1e:1c  10     full
0       0:07:38:49  221      27
0    ge0/6      57.0.1.15/24        Up      Up      null   service    1500  00:0c:29:7d:1e:3a  10     full
0       0:07:38:49  196      3
0    ge0/7      10.0.100.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:44  10     full
0       0:07:44:47  783      497
0    loopback1  172.16.255.25/32    Up      Up      null   transport  1500  00:00:00:00:00:00  10     full
0       0:00:00:20  0        0
0    system     172.16.255.15/32    Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:07:38:25  0        0
1    ge0/4      10.20.24.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:26  10     full
0       0:07:38:46  27594    27405
1    ge0/5      56.0.1.15/24        Up      Up      null   service    1500  00:0c:29:7d:1e:30  10     full
0       0:07:38:46  196      2
512  eth0       10.0.1.15/24        Up      Up      null   service    1500  00:50:56:00:01:05  1000   full
0       0:07:45:55  15053    10333
```

To allow vEdge routers at different overlay network sites on the private network to exchange data traffic directly, you use a loopback interface on the each vEdge router to handle the data traffic instead of using the physical interface that connects to the WAN. You associate the same tag, called a carrier tag, with each loopback interface so that all the routers learn that they are on the same private WAN. Because the loopback interfaces are advertised across the overlay network, the vEdge routers are able to learn reachability information, and they can exchange data traffic over the private network. To allow the data traffic to actually be transmitted out the WAN interface, you bind the loopback interface to a physical WAN interface, specifically to the interface that connects to the private network. Remember that this is the interface that the private network does not advertise. However, it is still capable of transmitting data traffic.

Exchange Data Traffic between Two Private WANs

This example shows a topology with two different private networks, possibly the networks of two different network providers, and all the Viptela devices are located somewhere on one or both of the private networks. Two vEdge routers are located at two different sites, and they both connect to both private networks. A third vEdge router connects to only one of the private WANs. The vBond orchestrator and vSmart controller both sit in one of the private WANs, perhaps in a data center, and they are reachable over both private WANs. For the vEdge routers to be able to establish control connections, the subnetworks where the vBond and vSmart devices reside must be advertised into each private WAN. Each private WAN CPE router then advertises these subnets in its VRF, and each vEdge router learns those prefixes from each PE router that it is connected to.



Because both WANs are private, we need only a single overlay. In this overlay network, without policy, IPsec tunnels running BFD sessions exist from any TLOC connected to either transport network to any TLOC in the other transport as well as to any TLOC in the same WAN transport network.

As with the previous examples in this article, it is possible to configure the tunnel interfaces on the routers' physical interfaces. If you do this, you also need to configure a routing protocol between the vEdge router at its peer PE router, and you need to configure access lists on the vEdge router to advertise all the routes in both private networks.

A simpler configuration option that avoids the need for access lists is to use loopback interfaces as the tunnel interfaces, and then bind each loopback interface to the physical interface that connects to the private network. Here, the loopback interfaces become the end points of the tunnel, and the TLOC connections in the overlay network run between loopback interfaces, not between physical interfaces. So in the figure shown above, on router vEdge-1, the tunnel connections originate at the Loopback1 and Loopback2 interfaces. This router has two TLOCs: {1.1.1.1, private2, ipsec} and {1.1.1.2, private1, ipsec}.

The WAN interfaces on the vEdge routers must run a routing protocol with their peer PE routers. The routing protocol must advertise the vEdge router's loopback addresses to both PE routers so that all vEdge routers on the two private networks can learn routes to each other. A simple way to advertise the loopback addresses is to redistribute routes learned from other (connected) interfaces on the same router. (You do this instead of creating access lists.) If, for example, you are using OSPF, you can advertise the loopback addresses by including the

redistribute connected command in the OSPF configuration. Looking at the figure above, the **ge0/2** interface on vEdge-1 needs to advertise both the Loopback1 and Loopback2 interfaces to the blue private WAN, and **ge0/1** must advertise also advertise both these loopback interfaces to the green private WAN.

With this configuration:

- The vEdge routers learn the routes to the vBond orchestrator and vSmart controller over each private WAN transport.
- The vEdge routers learn every other vEdge router's loopback address over each WAN transport network.
- The end points of the tunnel connections between each pair of vEdge routers are the loopback interfaces, not the physical (**ge**) interfaces.
- The overlay network has data plane connectivity between any TLOCs and has a control plane over both transport networks.

Here is the interface configuration for VPN 0 on vEdge-1. Highlighted are the commands that bind the loopback interfaces to their physical interfaces. Notice that the tunnel interfaces, and the basic tunnel interface properties (encapsulation and color), are configured on the loopback interfaces, not on the Gigabit Ethernet interfaces.

```
vpn 0
  interface loopback1
    ip address 1.1.1.2/32
    tunnel-interface
      encapsulation ipsec
      color private1
      bind ge0/1
    !
    no shutdown
  !
  interface loopback2
    ip address 1.1.1.1/32
    tunnel-interface
      encapsulation ipsec
      color private2
      bind ge0/2
    !
    no shutdown
  !
  interface ge0/1
    ip address 172.16.13.3/24
    no shutdown
  !
  interface ge0/2
    ip address 10.10.23.3/24
    no shutdown
  !
  ip route 0.0.0.0/0 10.10.23.1
  ip route 0.0.0.0/0 172.16.13.1
!
```

The configuration for vEdge-2 is similar:

```
vpn 0
  interface loopback1
    ip address 2.2.2.1/32
    tunnel-interface
      encapsulation ipsec
      color private1
      bind ge0/1
    !
    no shutdown
  !
  interface loopback2
    ip address 2.2.2.2/32
    tunnel-interface
```

```
      encapsulation ipsec
      color private2
      bind ge0/2
    !
    no shutdown
  !
  interface ge0/1
    ip address 172.16.15.5/24
    no shutdown
  !
  interface ge0/2
    ip address 10.10.25.3/24
    no shutdown
  !
  ip route 0.0.0.0/0 10.10.25.1
  ip route 0.0.0.0/0 172.16.15.1
!
```

The vEdge-3 router connects only to the green private WAN:

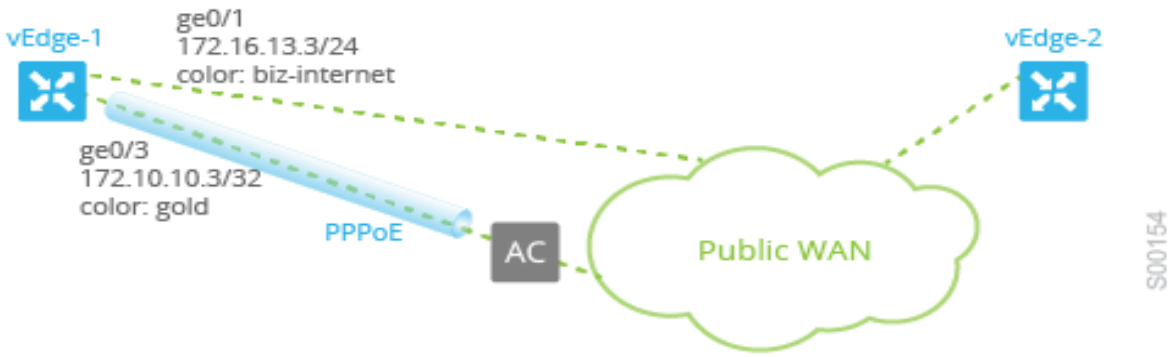
```
vpn 0
  interface loopback1
    ip address 3.3.3.3/32
    tunnel-interface
      encapsulation ipsec
      color private1
      bind ge0/1
    !
    no shutdown
  !
  interface ge0/1
    ip address 172.16.8.4/24
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.8.1
!
```

On the vSmart controller and vBond orchestrator, you configure a tunnel interface and default IP route to reach the WAN transport. For the tunnel, color has no meaning because these devices have no TLOCs.

```
vpn 0
  interface eth1
    ip address 172.16.8.9/24
      tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.8.1
!
vpn 0
  interface ge0/1
    ip address 172.16.16.6/24
      tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.16.1
!
```

Connect to a WAN Using PPPoE

This example shows a vEdge router with a TLOC tunnel interface and an interface enabled for Point-to-Point Protocol over Ethernet (PPPoE). The PPP interface defines the authentication method and credentials and is linked to the PPPoE-enabled interface.



Here is the interface configuration for VPN 0:

```
vpn 0
 interface ge0/1
  no shutdown
  !
  tunnel-interface
   encapsulation ipsec
   color biz-internet
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
 interface ge0/3
  pppoe-client ppp-interface ppp10
  no shutdown
 !
 interface ppp10
  ppp authentication chap
   hostname branch100@corp.bank.myisp.net
   password $4$OHHjdmsC6M8zj4BgLEFXKw==
  !
  tunnel-interface
   encapsulation ipsec
   color gold
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
!
```

Use the **show ppp interface** command to view existing PPP interfaces:

```
vEdge# show ppp interface

              PPPOE      INTERFACE               PRIMARY  SECONDARY
VPN  IFNAME   INTERFACE  IP         GATEWAY IP   DNS      DNS        MTU
-------------------------------------------------------------------------
0    ppp10    ge0/3      11.1.1.1   115.0.1.100  8.8.8.8  8.8.4.4    1150
```

Use the **show ppppoe session** and **show pppoe statistics** commands to view information about PPPoE sessions:

```
vEdge# show pppoe session

             SESSION                                          PPP                    SERVICE
VPN  IFNAME  ID       SERVER MAC         LOCAL MAC           INTERFACE  AC NAME     NAME
-------------------------------------------------------------------------------------------
0    ge0/1   1        00:0c:29:2e:20:1a  00:0c:29:be:27:f5   ppp1       branch100   -
0    ge0/3   1        00:0c:29:2e:20:24  00:0c:29:be:27:13   ppp2       branch100   -

vEdge# show pppoe statistics

      pppoe_tx_pkts             :      73
      pppoe_rx_pkts             :      39
      pppoe_tx_session_drops    :      0
      pppoe_rx_session_drops    :      0
      pppoe_inv_discovery_pkts  :      0
      pppoe_ccp_pkts            :      12
      pppoe_ipcp_pkts           :      16
      pppoe_lcp_pkts            :      35
      pppoe_padi_pkts           :      4
      pppoe_pado_pkts           :      2
      pppoe_padr_pkts           :      2
      pppoe_pads_pkts           :      2
      pppoe_padt_pkts           :      2
```

Additional Information

Configuring Interfaces
Configuring Segmentation (VPNs)

Configuring System Logging

On Viptela devices, you can log event notification system log (syslog) messages to files on the local device, or you can log them to files on a remote host.

Log Syslog Messages on the Local Device

Logging to the local device's hard disk of syslog messages with a priority level of "information" is enabled by default. The log files are placed in the local /var/log directory. By default, log files are 10 MB in size, and up to 10 files are stored. After 10 files have been created, the oldest one is discarded to create a file for newer syslog messages.

To modify the syslog default parameters on a Viptela device:

```
Viptela(config)# system logging disk
Viptela(config-logging-disk)# enable
Viptela(config-logging-disk)# file rotate number size megabytes
Viptela(config-logging-disk)# priority priority
```

By default, 10 syslog files are created. In the rotate command, you can configure this to be a number from 1 through 10.

By default, syslog files are 10 MB. You can configure this to be from 1 to 20 MB.

The priority indicates the severity of syslog messages to save. The default priority value is "informational", so by default, all syslog messages are recorded. The priority level can be one of the following (in order of decreasing severity):

- **emergency** —System is unusable (corresponds to syslog severity 0).
- **alert** — Action must be taken immediately (corresponds to syslog severity 1).
- **critical** —A serious condition (corresponds to syslog severity 2).

- **error** —An error condition that does not fully impair system usability (corresponds to syslog severity 3).
- **warn** —A minor error condition (corresponds to syslog severity 4).
- **normal** —A normal, but significant condition (corresponds to syslog severity 5).
- **information** —Routine condition (the default) (corresponds to syslog severity 6).

To disable the logging of syslog messages to the local disk, use the **no system logging disk enable** command.

Log Syslog Messages to a Remote Device

To log event notification syslog messages to a remote host, configure information about the server:

```
Viptela(config)# system logging server (dns-name | hostname | ip-address)
Viptela(config-logging-server)# vpn vpn-id
Viptela(config-logging-server)# priority priority
Viptela(config-logging-server)# source-interface interface-name
```

Configure the server's name by DNS name, hostname, or IP address. You can configure up to four syslog servers.

You can optionally specify the VPN in which the syslog server is located or through which it can be reached.

You can optionally specify the outgoing interface to use to reach the syslog server. The interface name can be a physical interface or a subinterface (a VLAN-tagged interface). The interface must be located in the same VPN as the syslog server. Otherwise, the configuration is ignored. If you configure multiple syslog servers, the source interface must be the same for all of them.

You configure the priority of the syslog messages to send to the server, as described above.

Viptela devices send syslog messages to syslog servers using UDP. TCP is not supported.

If the syslog server is unreachable, the Viptela device suspends the sending of syslog messages for 180 seconds (3 minutes). If the server is once again reachable, logging resumes. If not, the Viptela device waits another 180 seconds.

By default, syslog messages are also always logged to the local hard disk. To disable local logging, use the **no system logging disk enable** command.

Display Logging Information

To display the configured system logging settings, use the **show logging** command. For example:

```
vEdge# show logging

System logging to  in vpn 0 is disabled
Priority for host logging is set to: info

System logging to disk is enabled
Priority for disk logging is set to:  info
File name for disk logging is set to:  /var/log/vsyslog
File size for disk logging is set to:  10 MB
File recycle count for disk logging is set to:  10

Syslog facility is set to: local7
```

To display the contents of a syslog file, use the **show log** command. For example:

```
vEdge# show log auth.log tail 10
==> /var/log/auth.log <==
auth.info: Nov 14 14:33:35 vedge sshd[2570]: Accepted publickey for admin from 10.0.1.1 port 39966 ssh2:
RSA SHA256:pkFQ5wE//DmiA0d0JU1rOt91CMTVGkscm9wLSYQrI1s
auth.info: Nov 14 14:39:42 vedge sshd[2578]: Received disconnect from 10.0.1.1 port 39966:11:
```

```
disconnected by user
auth.info: Nov 14 14:39:42 vedge sshd[2578]: Disconnected from 10.0.1.1 port 39966
auth.info: Nov 16 10:51:45 vedge sshd[6106]: Accepted publickey for admin from 10.0.1.1 port 40012 ssh2:
RSA SHA256:pkFQ5wE//DmiA0d0JU1rOt91CMTVGkscm9wLSYQrI1s
auth.info: Nov 16 11:21:55 vedge sshd[6108]: Received disconnect from 10.0.1.1 port 40012:11:
disconnected by user
auth.info: Nov 16 11:21:55 vedge sshd[6108]: Disconnected from 10.0.1.1 port 40012
auth.info: Nov 17 12:59:52 vedge sshd[15889]: Accepted publickey for admin from 10.0.1.1 port 40038 ssh2:
RSA SHA256:pkFQ5wE//DmiA0d0JU1rOt91CMTVGkscm9wLSYQrI1s
auth.info: Nov 17 13:45:13 vedge sshd[15894]: Received disconnect from 10.0.1.1 port 40038:11:
disconnected by user
auth.info: Nov 17 13:45:13 vedge sshd[15894]: Disconnected from 10.0.1.1 port 40038
auth.info: Nov 17 14:47:31 vedge sshd[30883]: Accepted publickey for admin from 10.0.1.1 port 40040 ssh2:
RSA SHA256:pkFQ5wE//DmiA0d0JU1rOt91CMTVGkscm9wLSYQrI1s
```

In vManage NMS, to display device syslog files to help in debugging:

1. In the Administration ► Settings screen, ensure that Data Stream is enabled.
2. From the Monitor ► Network screen, select the vEdge router.
3. Click Troubleshooting in the left pane.
4. From the Logs pane, click Debug Log.
5. In the Log Files field, select the name of the log file. The lower part of the screen displays the log information.

System Log Files

Syslog messages at or above the default or configured priority value are recorded in a number of files in the /var/log directory on the local device. These files include the following:

- auth.log—Login, logout, and superuser access events, and usage of authorization systems.
- kern.log—Kernel messages.
- messages—Consolidated log file that contains syslog messages from all sources.
- vconfd—All configuration-related syslog messages.
- vdebug—All debug messages for modules whose debugging is turned on and all syslog messages above the configured priority value. Debug logging supports various levels of logging based on the module. Different modules implement the logging levels differently. For example, the system manager (sysmgr) has two logging levels (on and off), while the chassis manager (chmgr) has four different logging levels (off, low, normal, and high). You cannot send debug messages to a remote host. To enable debugging, use the **debug** operational command.
- vsyslog—All syslog messages from Viptela processes (daemons) above the configured priority value. The default priority value is "informational", so by default, all "notice", "warning", "error", "critical", "alert", and "emergency" syslog messages are saved.

The Viptela software does not use the following standard LINUX files, which are present in /var/log, for logging: cron.log, debug, lpr.log, mail.log, and syslog.

The writing of messages to syslog files is not rate-limited. This means that if many syslog messages are generated in a short amount of time, the overflow messages are buffered and placed in a queue until they can be written to a syslog file. The overflow messages are not dropped.

For repeating syslog messages—identical messages that occur multiple times in succession—only one copy of the message is placed in the syslog file. The message is annotated to indicate the number of times that the message occurred.

The maximum length of a log message is 1024 bytes. Longer messages are truncated.

Syslog messages related to AAA authentication and Netconf CLI access and usage are placed in the auth.log and messages files. Each time a vManage NMS logs in to a vEdge router to retrieve statistics and status information and to push files to the router, the router generates AAA and Netconf log messages. So, over time, these message can fill the log files. To prevent these messages from filling the log files, you can disable the logging of AAA and Netconf syslog messages:

```
Viptela(config)# system aaa logs
Viptela(config-logs)# audit-disable
Viptela(config-logs)# netconf-disable
```

Syslog message generated by the Viptela software have the following format:

```
facility.source date - source - module - level - MessageID: text-of-syslog-message
```

Here is an example of a syslog message. In the file, this message is on a single line. This message has the facility name of local7, which is the name used for all Viptela processes, and a priority of "info".

```
local7.info: Apr  3 13:40:31 vsmart SYSMGR[221]:
%Viptela-vsmart-sysmgrd-6-INFO-1400002:
Notification : 4/3/2017 20:40:31 system-login-change severity-level:minor host-name:"vm1" system-
ip:172.16.255.11 user-name:"admin" user-id:162
```

Additional Information

show log
show logging
Use Syslog Messages

Configuring IEEE 802.1X and IEEE 802.11i Authentication

IEEE 802.1X is a port-based network access control (PNAC) protocol that prevents unauthorized network devices from gaining access to wired networks (WANs), by providing authentication for devices that want to connect to a WAN.

IEEE 802.11i prevents unauthorized network devices from gaining access to wireless networks (WLANs). 802.11i implements WiFi Protected Access II (WPA2) to provide authentication for devices that want to connect to a WLAN on a vEdge 100wm router.

A RADIUS authentication server must authenticate each client connected to a port before that client can access any services offered by network.

This article describes how to configure RADIUS servers to use for 802.1X and 802.11i authentication. It describes how to enable 802.1X on vEdge router interfaces to have the router act as an 802.1X authenticator, responsible for authorizing or denying access to network devices on a WAN. Finally, it describes how to enable 802.11i on vEdge 100wm routers to control access to WLANs.

Configure RADIUS Authentication Servers

Authentication services for IEEE 802.1X and IEEE 802.11i are provided by RADIUS authentication servers. You configure the RADIUS servers to use for 802.1X and 802.11i authentication on a system-wide basis:

```
vEdge(config)# system radius
vEdge(config-radius)# server ip-address
```

Specify the IP address of the RADIUS server. You can configure one or two RADIUS servers to perform 802.1X and 802.11i authentication. (Note that for AAA authentication, you can configure up to eight RADIUS servers.)

For each RADIUS server, you can configure a number of optional parameters.

You can configure the VPN through which the RADIUS server is reachable and the router interface to use to reach the server:

```
vEdge(config-server)# vpn vpn-id
vEdge(config-server)# source-interface interface-name
```

If you configure two RADIUS servers, they must both be in the same VPN, and they must both be reachable using the same source interface.

You must configure a tag to identify the RADIUS server:

```
vEdge(config-server)# tag tag
```

The tag can be from 4 through 16 characters. You use this tag when configuring the RADIUS servers to use with IEEE 802.1X authentication and with IEEE 802.11i WPA enterprise authentication (discussed later in this article).

For authentication between the router and the RADIUS server, you can authenticate and encrypt packets sent between the vEdge router and the RADIUS server, and you can configure a destination port for authentication requests. To authenticate and encrypt packets, configure a key:

```
vEdge(config-server)# secret-key password
```

Enter the password as clear text, which is immediately encrypted, or as an AES 128-bit encrypted key. The key must match the AES encryption key used on the RADIUS server.

By default, UDP port 1812 is used as the destination port on the RADIUS server to use for authentication requests. You can change the port number to a number from 1 through 65535. To disable authentication, set the port number to 0.

```
vEdge(config-server)# auth-port number
```

You can set the priority of a RADIUS server, to choose which one to use first when performing 802.1X authentication:

```
vEdge(config-server)# priority number
```

The priority can be a value from 0 through 7. The server with the lower priority number is given priority. If you do not include this command in the RADIUS server configuration, the priority is determined by the order in which you enter the IP addresses in the **system radius server** command.

By default, accounting in enabled for 802.1X and 802.11i interfaces. Accounting information is sent to UDP port 1813 on the RADIUS server. To change this port:

```
vEdge(config-server)# acct-port number
```

The port number can be from 1 through 65535.

Configure IEEE 802.1X Port Security

To enable basic 802.1X port security on an interface, configure it and at least one RADIUS server to use for 802.1X authentication. The 802.1X interface must be in VPN 0.

```
vEdge(config)# vpn 0 interface interface-name
vEdge(config-interface)# dot1x
vEdge(config-dot1x)# radius-servers tag
```

For 802.1X authentication to work, you must also configure the same interface under an untagged bridge:

```
vEdge(config)# bridge number
vEdge(config)# interface interface-name
```

The interface name in the **vpn 0 interface** and **bridge interface** commands must be the same. Do not configure a VLAN ID for this bridge so that it remains untagged.

You can enable 802.1X on a maximum of four wired physical interfaces. The interface cannot also be configured as a tunnel interface.

Configure the tags associated with one or two RADIUS servers to use for 802.1X client authentication and accounting. (You configure the tags with the **system radius server tag** command.) If you specify tags for two RADIUS servers, they must both be reachable in the same VPN. If you do not configure a priority value when you configure the RADIUS server with the **system radius server priority** command, the order in which you list the IP addresses is the order in which the RADIUS servers are tried.

Enable RADIUS Accounting

By default, the vEdge router never sends interim accounting updates to the 802.1X RADIUS accounting server. Accounting updates are sent only when the 802.1X session ends.

To enable the sending of interim accounting updates, configure the interval at which to send the updates:

```
vEdge(config-dot1x)# accounting-interval seconds
```

The time can be from 0 through 7200 seconds.

Enable MAC Authentication Bypass

IEEE 802.1X authentication is accomplished through an exchange of Extensible Authentication Procotol (EAP) packets. After 802.1X-compliant clients respond to the EAP packets, they can be authenticated and granted access to the network. Enabling MAC authentication bypass (MAB) provides a mechanism to allow non-802.1X–compliant clients to be authenticated and granted access to the network.

The vEdge router determines that a device is non-802.1X–compliant clients when the 802.1X authentication process times out while waiting for an EAPOL response from the client.

To enable MAC authentication bypass for an 802.1X interface on the vEdge router:

```
vEdge(config)# vpn 0 interface interface-name dot1x
vEdge(config-dot1x)#  mac-authentication-bypass
```

With this configuration, the vEdge router authenticates non-802.1X–compliant clients using the configured RADIUS servers. The RADIUS server must be configured with the MAC addresses of non-802.1X–compliant clients that are allowed to access the network.

To enable MAB on the RADIUS server:

```
vEdge(config-dot1x)# mac-authentication-bypass server
```

To allow authentication to be performed for one or more non-802.1X–compliant clients before performing an authentication check with the RADIUS server, list their MAC addresses in the following command:

```
vEdge(config-dot1x)# mac-authentication-bypass allow mac-addresses
```

You can configure up to eight MAC addresses for MAC authentication bypass. For these devices, the vEdge router grants immediate network access based on their MAC addresses, and then sends a request to the RADIUS server to authenticate the devices.

Configure VLANs for Authenticated and Unauthenticated Clients

For clients that cannot be authenticated but that you want to provide limited network services to, you create VLANs to handle network access for these clients. You also create VLANs to handle authenticated clients.

You can create the following kinds of VLAN:

- Guest VLAN—Provide limited services to non-802.1X–compliant clients.
- Authentication Reject VLAN—Provide limited services to 802.1X-compliant clients that failed RADIUS authentication. An authentication-reject VLAN is similar to a restricted VLAN.
- Authentication Fail VLAN—Provide network access when RADIUS authentication or the RADIUS server fails. An authentication-fail VLAN is similar to a critical VLAN.

- Default VLAN—Provide network access to 802.1X–compliant clients that are successfully authenticated by the RADIUS server. If you do not configure a default VLAN on the vEdge router, successfully authenticated clients are placed into VLAN 0, which is the VLAN associated with an untagged bridge.

To configure the VLANs for authenticated and unauthenticated clients, first create the VLAN in a bridging domain, and then create the 802.1X VLANs for the unauthenticated clients by associating the bridging domain VLAN with an 802.1X VLAN.

To create the VLAN, configure a bridging domain to contain the VLAN:

```
vEdge(config)# bridge bridge-id
vEdge(config-bridge)# name text
vEdge(config-bridge)# vlan vlan-id
vEdge(config-bridge)# interface interface-name
vEdge(config-interface)# no shutdown
```

The bridging domain identifier is a number from 1 through 63. A best practice is to have the bridge domain ID be the same as the VLAN number.

The name is optional, but it is recommended that you configure a name that identifies the 802.1X VLAN type, such as Guest-VLAN and Default-VLAN.

The VLAN number can be from 1 through 4095. This is the number that you associate with an 802.1X VLAN.

The interface name is the interface that is running 802.1X.

Then configure the 802.1X VLANs to handle unauthenticated clients.

A guest VLAN provides limited services to non-802.1X–compliant clients, and it can be used to allow clients to download 802.1X client software. An interface running 802.1X assigns clients to a guest VLAN when the interface does not receive a response to EAP request/identity packets that it has sent to the client, or when the client does not send EAPOL packets and MAC authentication bypass is not enabled. To configure a guest VLAN:

```
vEdge(config)# vpn 0 interface interface-name interface dot1x
vEdge(config-dot1x)# guest-vlan vlan-id
```

The VLAN number must match one of the VLANs you configured in a bridging domain. A best practice is to have the VLAN number be the same as the bridge domain ID.

An authentication-reject VLAN provides limited services to 802.1X-compliant clients that have failed RADIUS authentication. To configure an authentication-reject VLAN:

```
vEdge(config-dot1x)# auth-reject-vlan vlan-id
```

The VLAN number must match one of the VLANs you configure in a bridging domain. A best practice is to have the VLAN number be the same as the bridge domain ID.

When the RADIUS authentication server is not available, 802.1X-compliant clients attempting to authenticate are placed in an authentication-fail VLAN if it is configured. If this VLAN is not configured, the authentication request is eventually dropped. To configure the authentication-fail VLAN:

```
vEdge(config-dot1x)# auth-fail-vlan vlan-id
```

The VLAN number must match one of the VLANs you configure in a bridging domain. A best practice is to have the VLAN number be the same as the bridge domain ID.

The following configuration snippet illustrates the interrelationship between the 802.1X configuration and the bridging domain configuration. This snippet shows that the bridging domain numbers match the VLAN numbers, which is a recommended best practice. Also, the bridging domain name identifies the type of 802.1X VLAN.

```
system
...
 radius
  server 10.1.15.150
   tag              freerad1
```

```
    source-interface ge0/0
    secret-key       $4$L3rwZmsIic8zj4BgLEFXKw==
    priority         1
   exit
   server 10.20.24.150
    auth-port        2000
    acct-port        2001
    tag              freerad2
    source-interface ge0/4
    secret-key       $4$L3rwZmsIic8zj4BgLEFXKw==
    priority         2
   exit
  !
 !
 bridge 1
  name Untagged_bridge
  interface ge0/5
   no native-vlan
   no shutdown
  !
 !
 bridge 10
  name Authorize_VLAN
  vlan 10
  interface ge0/5
   no native-vlan
   no shutdown
  !
 !
 bridge 20
  name Guest_VLAN
  vlan 20
  interface ge0/5
   no native-vlan
   no shutdown
  !
 !
 bridge 30
  name Critical_VLAN
  vlan 30
  interface ge0/5
   no native-vlan
   no shutdown
  !
 !
 bridge 40
  name Restricted_VLAN
  vlan 40
  interface ge0/5
   no native-vlan
   no shutdown
  !
 !
 vpn 0
  interface ge0/0
   ip address 10.1.15.15/24
   tunnel-interface
    encapsulation ipsec
  ...
   !
   no shutdown
  !
  interface ge0/1
   ip address 60.0.1.16/24
   no shutdown
```

```
 !
 interface ge0/2
  ip address 10.1.19.15/24
  no shutdown
 !
 interface ge0/4
  ip address 10.20.24.15/24
  no shutdown
 !
 interface ge0/5
  dot1x
   auth-reject-vlan 40
   auth-fail-vlan   30
   guest-vlan       20
   default-vlan     10
   radius-servers   freerad1
  !
  no shutdown
 !
 interface ge0/7
  ip address 10.0.100.15/24
  no shutdown
 !
!
vpn 1
 interface ge0/2.1
  ip address 10.2.19.15/24
  mtu      1496
  no shutdown
 !
 interface irb1
  ip address 56.0.1.15/24
  mac-address 00:00:00:00:aa:01
  no shutdown
  dhcp-server
   address-pool 56.0.1.0/25
   offer-time   600
   lease-time   86400
   admin-state  up
   options
    default-gateway 56.0.1.15
   !
  !
 !
!
vpn 10
 interface ge0/2.10
  ip address 10.10.19.15/24
  mtu      1496
  no shutdown
 !
 interface irb10
  ip address 56.0.10.15/24
  mac-address 00:00:00:00:aa:10
  no shutdown
  dhcp-server
   address-pool 56.0.10.0/25
   offer-time   600
   lease-time   86400
   admin-state  up
   options
    default-gateway 56.0.10.15
   !
  !
```

```
 !
!
vpn 20
 interface ge0/2.20
  ip address 10.20.19.15/24
  mtu      1496
  no shutdown
 !
 interface irb20
  ip address 56.0.20.15/24
  mac-address 00:00:00:00:aa:20
  no shutdown
 !
!
vpn 30
 interface ge0/2.30
  ip address 10.30.19.15/24
  mtu      1496
  no shutdown
 !
 interface irb30
  ip address 56.0.30.15/24
  mac-address 00:00:00:00:aa:30
  no shutdown
 !
!
vpn 40
 interface ge0/2.40
  ip address 10.40.19.15/24
  mtu      1496
  no shutdown
 !
 interface irb40
  ip address 56.0.40.15/24
  mac-address 00:00:00:00:aa:40
  no shutdown
 !
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
 !
!
```

Configure Control Direction

To configure how the 802.1X interface handles traffic when the client is unauthorized, set the control direction:

```
vEdge(config-dot1x)# control-direction (in-and-out | in-only)
```

The direction can be one of the following:

- **in-and-out** —The 802.1X interface can both send packets to and receive packets from the authorized client. Bidirectional control is the default behavior.
- **in-only** —The 802.1X interface can send packets to the unauthorized client, but cannot receive packets from that client.

Configure Authentication with Wake on LAN

IEEE 802.1X authentication wake on LAN (WoL) allows dormant clients to be powered up when the vEdge router receives a type of Ethernet frame called the magic packet. Administrators can use wake on LAN when to connect to systems that have been powered down.

When a client that uses wake on LAN and that attaches through an 802.1X port powers off, the 802.1X port becomes unauthorized. The port can only receive and send EAPOL packets, and wake-on-LAN magic packets cannot reach the client. When the device is powered off, it is not authorized, and the switch port is not opened.

Without wake on LAN, when an 802.1X port is unauthorized, the router's 802.1X interface block traffic other than EAPOL packets coming from unauthorized clients.

When you enable wake on LAN on an 802.1X port, the vEdge router is able to send magic packets even if the 802.1X port is unauthorized.

To enable wake on LAN on an 802.1X interface, use the following command:

```
vEdge(config)# vpn 0 interface interface-name dot1x
vEdge(config-dot1x)# wake-on-lan
```

Configure 802.1X Host Mode

The host mode of an 802.1X interfaces determines whether the interface grants access to a single client or to multiple clients. Three host modes are available:

- Single-host mode—The 802.1X interface grants access only to the first authenticated client. All other clients attempting access are denied and dropped.
- Multiple-host mode—A single 802.1X interface grants access to multiple clients. In this mode, only one of the attached clients must be authorized for the interface to grant access to all clients. If the interface becomes unauthorized, the vEdge router denies network access to all the attached clients.
- Multiple-authentication mode—A single 802.1X interface grants access to multiple authenticated clients on data VLANs.

To configure the host mode of the 802.1X interface, use the following command:

```
vEdge(config)# vpn 0 interface interface-name dot1x
vEdge(config-dot1x)# host-mode (multi-auth | multi-host | single-host)
```

Set the Timeout for Inactive Clients

By default, when a client has been inactive on the network for 1 hour, its authentication is revoked, and the client is timed out. To change the timeout interval, use the following command:

```
vEdge(config)# vpn 0 interface interface-name dot1x
vEdge(config-dot1x)# timeout inactivity minutes
```

The timeout interval can be from 0 through 1440 minutes (24 hours).

Enable Periodic Client Reauthentication

By default, once a client session is authenticated, that session remains functional indefinitely. To enable the periodic reauthentication of 802.1X clients, configure the number of minutes between reauthentication attempts:

```
vEdge(config)# vpn 0 interface interface-name dot1x
vEdge(config-dot1x)# reauthentication minutes
```

The time can be from 0 through 1440 minutes (24 hours)

Configure Dynamic Authorization Service for RADIUS Change of Authorization

Dynamic authorization service (DAS) allows an 802.1X interface on a vEdge router to accept change of authorization (CoA) requests from a RADIUS or other authentication server and to act on the requests. The Viptela implementation of DAS supports disconnect packets, which immediately terminate user sessions, and reauthentication CoA requests, which modify session authorization attributes.

DAS, defined in RFC 5176 , is an extension to RADIUS that allows the RADIUS server to dynamically change 802.1X session information without requiring the vEdge router to initiate the change request. When you enable DAS on the vEdge router, the router opens a socket to listen for CoA requests from the RADIUS server. If the network administrator of a RADIUS server modifies the authentication of an 802.1X client, the RADIUS server sends a CoA request to inform the router about the change of authorization. When the router receives the CoA request, it processes the requested change.

To enable DAS for an 802.1X interface, you configure information about the RADIUS server from which the interface can accept CoA requests. In the context of configuring DAS, the vEdge router is the server and the RADIUS server (or other authentication server) is the client.

To configure the RADIUS server from which to accept CoA requests, configure the server's IP address and the password that the RADIUS server uses to access the router's 802.1X interface:

```
vEdge(config)# vpn 0 interface interface-name dot1x
vEdge(config-dot1x)# das
vEdge(config-das)# client ip-address
vEdge(config-das)# secret-key password
```

You can configure the VPN through which the RADIUS server is reachable:

```
vEdge(config-das)# vpn vpn-id
```

By default, the 802.1X interface uses UDP port 3799 to listen for CoA request from the RADIUS server. You can change the port number:

```
vEdge(config-das)# port port-number
```

The port number can be a value from 1 through 65535. If you configure DAS on multiple 802.1X interfaces on a vEdge router, you must configure each interface to use a different UDP port.

By default, the CoA requests that the vEdge router receives from the DAS client are all honored, regardless of when the router receives them. To have the router handle CoA within a specified time, you require that the DAS client timestamp all CoA requests:

```
vEdge(config-das)# require-timestamp
```

With this configuration, the vEdge router processes only CoA requests that include an event timestamp. Non-timestamped CoA requests are dropped immediately.

When timestamping is configured, both the vEdge router and the RADIUS server check that the timestamp in the CoA request is current and within a specific time window. The default time window is 300 seconds (5 minutes). This behavior means that if the DAS timestamps a CoA at 15:00 and the router receives it at 15:04, the router honors the request. However, if the router receives the request at 15:10, the router drops the CoA request. You can change the time window to a time from 0 through 1000 seconds:

```
vEdge(config-das)# time-window seconds
```

Configure RADIUS Authentication and Accounting Attributes

For IEEE 802.1X authentication and accounting, the vEdge router, acting as a network access server (NAS), sends RADIUS attribute–value (AV) pairs to the RADIUS server. These AV pairs are defined in RFC 2865 , RADIUS, RFC 2866 , RADIUS Accounting, and RFC 2869 , RADIUS Extensions. The AV pairs are placed in the Attributes field of the RADIUS packet.

By default, when you enable IEEE 802.1X port security, the following authentication attributes are included in messages sent to the RADIUS server:

Attribute Number	Attribute Name	Description
1	User-Name	Name of the user to be authenticated.
5	NAS-Port	Physical port number on the vEdge router that is authenticating the user.
12	Framed-MTU	Maximum MTU configured for the user.
30	Called-Station-Id	Phone number that the user called, using dialed number identification (DNIS) or similar technology used to access the RADIUS server.
31	Calling-Station-Id	Phone number that the call came in to the server, using automatic number identification (ANI) or similar technology.
44	Acct-Session-Id	Unique session identifier.
61	NAS-Port-Type	Type of physical port on the vEdge router that is authenticating the user.
77	Connect-Info	Nature of the user's connection.
79	EAP-Message	Encapsulate Extended Access Protocol (EAP) packets, to allow the vEdge router to authenticate dial-in users via EAP without having to run EAP.
80	Message-Authenticator	Sign RADIUS Access-Requests to prevent these requests from being spoofed by ARAP, CHAP, or EAP.

When you enable RADIUS accounting, the following accounting attributes are included, by default, in messages sent to the RADIUS server:

Attribute Number	Attribute Name	Description
1	User-Name	Name of the user to be authenticated.
5	NAS-Port	Physical port number on the vEdge router that is authenticating the user.
30	Called-Station-Id	Phone number that the user called, using dialed number identification (DNIS) or similar technology used to access the RADIUS server.
31	Calling-Station-Id	Phone number that the call came in to the server, using automatic number identification (ANI) or similar technology.
40	Acct-Status-Type	Mark the beginning and end of an accounting request.
44	Acct-Session-Id	Unique accounting identifier used to match the start and stop records in a log file.
45	Acct-Authentic	How the user was authenticated.
61	NAS-Port-Type	Type of physical port on the vEdge router that is authenticating the user.
77	Connect-Info	Nature of the user's connection.

Several configuration commands allow you to add additional attribute information to RADIUS packets.

To include the NAS-IP-Address (attribute 4) in messages sent to the RADIUS server, to indicate the IP address of the vEdge router that is acting as a NAS server:

```
vEdge(config-dot1x) nas-ip-address ip-address
```

To include the NAS-Identifier (attribute 32) in messages sent to the RADIUS server, use the following command:

```
vEdge(config-dot1x)# nas-identifier string
```

The NAS identifier is a unique string from 1 through 255 characters long that identifies the vEdge router that is acting as a NAS server.

To include a RADIUS authentication or accounting attribute of your choice in messages sent to the RADIUS server, use the following commands:

```
vEdge(config-dot1x)# auth-req-attr attribute-number (integer integer | octet octet | string string)
vEdge(config-dot1x)# acct-req-attr attribute-number (integer integer | octet octet | string string)
```

Specify the desired value of the attribute as an integer, octet value, or string, depending on the attribute. For example, to set the Service-Type attribute to be authenticate-only:

```
vEdge(config-dot1x)# auth-req-attr 6 integer 8
```

Configure IEEE 802.11i Authentication

For Viptela routers that support wireless LANs (WLANs), you can configure the router to support either a 2.4-GHz or 5-GHz radio frequency. Then, you segment the WLAN into multiple broadcast domains, which are called virtual access points, or VAPs. Users who connect to a VAP can be unauthenticated, or you can configure IEEE 802.11i authentication for each VAP.

For information about configuring the WLAN interface itself, see Configuring WLAN Interfaces .

To enable user authentication on the WLAN, you create a VAP on the desired radio frequency and then you configure Wi-Fi protected access (WPA) or WPA2 data protection and network access control for the VAP. WPA authenticates individual users on the WLAN using a username and password. WPA uses the Temporal Key Integrity Protocol (TKIP), which is based on the RC4 cipher. WPA2 implements the NIST FIPS 140-2–compliant AES encryption algorithm along with IEEE 802.1X-based authentication, to enhance user access security over WPA. WPA2 uses the Counter Mode Cipher Block Chaining Message Authentication Code Protocol (CCMP), which is based on the AES cipher. Authentication is done either using preshared keys or through RADIUS authentication.

To enable personal authentication, which requires users to enter a password to connect to the WLAN, configure the authentication and password:

```
vEdge(config)# wlan frequency
vEdge(config-wlan)# interface vapnumber
vEdge(config-vap)# no shutdown
vEdge(config-vap)# data-security (wpa-personal | wpa/wpa2-personal | wpa2-personal)
vEdge(config-vap)# wpa-personal-key password
```

For the security, configure either WPA, WPA2, or both (WPA/WPA2). Enter the password either as clear text or an AES-encrypted key.

For each VAP, you can customize the security mode to control wireless client access.

To enable enterprise WPA security, configure the authentication and the RADIUS server to perform the authentication:

```
vEdge(config-vap)# data-security (wpa-enterprise | wpa/wpa2-enterprise | wpa2-enterprise)
vEdge(config-vap)# radius-servers tag
```

For the security, configure either WPA, WPA2, or both (WPA/WPA2). Enter the password either as clear text or an AES-encrypted key.

In the **radius-servers** command, enter the tags associated with one or two RADIUS servers to use for 802.11i authentication. (You configure the tags with the **system radius server tag** command.) If you specify tags for two RADIUS servers, they must both be reachable in the same VPN. If you do not configure a priority value when you configure the RADIUS server with the **system radius server priority** command, the order in which you list the IP addresses is the order in which the RADIUS servers are tried.

By default, management frames sent on the WLAN are not encrypted. For each VAP, you can configure the encryption to be optional or required:

```
vEdge(config-vap)# mgmt-security (none | optional | required)
```

Additional Information

Configuring Network Interfaces
Configuring WLAN Interfaces

System, Interface, and SNMP CLI Reference

CLI commands for configuring and monitoring system-wide parameters, interfaces, and SNMP on vEdge routers and vSmart controllers.

Interface Configuration Commands

Use the following commands to configure interfaces and interface properties in the Viptela overlay network. You configure interfaces on a per-VPN basis.

```
vpn vpn-id
  interface interface-name
    access-list acl-list (on vEdge routers only)
    arp
      ip ip-address mac mac-address
    arp-timeout seconds (on vEdge routers only)
    autonegotiate (on vEdge routers only)
    block-non-source-ip (on vEdge routers only)
    clear-dont-fragment
    dead-peer-detection interval seconds retries number (on vEdge routers only)
    description text
    dhcp-helper ip-address (on vEdge routers only)
    dhcp-server (on vEdge routers only)
      address-pool prefix/length
      exclude ip-address
      lease-time seconds
      max-leases number
      offer-time minutes
      options
        default-gateway ip-address
        dns-servers ip-address
        domain-name domain-name
        interface-mtu mtu
        tftp-servers ip-address
      static-lease mac-address ip ip-address host-name hostname
    dot1x
      accounting-interval seconds
      acct-req-attr attribute-number (integer integer | octet octet | string string)
      auth-fail-vlan vlan-id
      auth-order (mab | radius)
      auth-reject-vlan vlan-id
      auth-req-attr attribute-number (integer integer | octet octet | string string)
      control-direction direction
      das
        client ip-address
        port port-number
        require-timestamp
        secret-key password
        time-window seconds
        vpn vpn-id
      default-vlan vlan-id
      guest-vlan vlan-id
      host-mode (multi-auth | multi-host | single-host)
      mac-authentication-bypass
        allow mac-addresses
        server
      nas-identifier string
      nas-ip-address ip-address
```

```
    radius-servers tag
    reauthentication minutes
    timeout
      inactivity minutes
    wake-on-lan
  duplex (full | half)
  flow-control (bidirectional | egress | ingress)
  ike (on vEdge routers only)
    authentication-type type
      local-id id
      pre-shared-secret password
      remote-id id
    cipher-suite suite
    group number
    mode mode
    rekey seconds
    version number
  (ip address prefix/length | ip dhcp-client [dhcp-distance number])
  (ipv6 address prefix/length | ipv6 dhcp-client [dhcp-distance number] [dhcp-rapid-commit])
  ip address-list prefix/length (on vSmart controller containers only)
  ip secondary-address ipv4-address (on vEdge routers only)
  ipsec (on vEdge routers only)
    cipher-suite suite
    perfect-forward-secrecy pfs-setting
    rekey seconds
    replay-window number
  keepalive seconds retries (on vEdge routers only)
  mac-address mac-address
  mtu bytes
  nat (on vEdge routers only)
    block-icmp-error
    block-icmp-error
    direction (inside | outside)
    [no] overload
    port-forward port-start port-number1 port-end port-number2
      proto (tcp | udp) private-ip-address ip address private-vpn vpn-id
    refresh (bi-directional | outbound)
    respond-to-ping
    static source-ip ip-address1 translate-ip ip-address2 (inside | outside)
    tcp-timeout minutes
    udp-timeout minutes
  pmtu (on vEdge routers only)
  policer policer-name (on vEdge routers only)
  ppp (on vEdge routers only)
    ac-name name
    authentication (chap | pap) hostname name password password
  pppoe-client (on vEdge routers only)
    ppp-interface name
  profile profile-id (on vEdge routers only)
  qos-map name (on vEdge routers only)
  rewrite-rule name (on vEdge routers only)
  shaping-rate name (on vEdge routers only)
  shutdown
  speed speed
  static-ingress-qos number (on vEdge routers only)
  tcp-mss-adjust bytes
  technology technology (on vEdge routers only)
  tloc-extension interface-name (on vEdge routers only)
  tracker tracker-name (on vEdge routers only)
  tunnel-interface
    allow-service service-name
    bind geslot/port (on vEdge routers only)
    carrier carrier-name
    color color [restrict]
    connections-limit number
```

```
      encapsulation (gre | ipsec) (on vEdge routers only)
        preference number
        weight number
      hello-interval milliseconds
      hello-tolerance seconds
      low-bandwidth-link (on vEdge routers only)
      max-control-connections number (on vEdge routers only)
      nat-refresh-interval seconds
      port-hop
      vbond-as-stun-server (on vEdge routers only)
      vmanage-connection-preference number (on vEdge routers only)
    tunnel-destination ip-address (GRE interfaces; on vEdge routers only)
    tunnel-destination (dns-name | ipv4-address) (IPsec interfaces; on vEdge routers only)
    (tunnel-source ip-address | tunnel-source-interface interface-name) (GRE interfaces; on vEdge routers
only)
    (tunnel-source ip-address | tunnel-source-interface interface-name) (IPsec interfaces; on vEdge
routers only)
    upgrade-confirm minutes
    vrrp group-name (on vEdge routers only)
      priority number
      timer seconds
      track-omp
```

Interface Monitoring Commands

Use the following commands to monitor interfaces:

show dhcp interface
show dhcp server
show interface
show interface arp-stats
show interface errors
show interface packet-sizes
show interface port-stats
show interface queue
show interface statistics
show vrrp

SNMP Configuration Commands

Use the following commands to configure SNMP:

```
snmp
  community name
    authorization (read-only | read-write)
    view string
  contact string
  group group-name authentication
    view string
  location string
  name string
  [no] shutdown
  trap
    group group-name
      trap-type
        level severity
    target vpn vpn-id ip-address udp-port
      community-name community-name
      group-name group-name
```

```
      source-interface interface-name
  user username
    auth authentication
    auth-password password
    group group-name
    priv privacy
    priv-password password
  view string
    oid oid-subtree [exclude]
```

SNMP Monitoring Commands

Use the following command to monitor SNMP:

show running-config snmp —Display the active configuration that is running on the Viptela device.

System Configuration Commands

Use the following commands to configure system-wide parameters:

```
banner
  login "text"
  motd "text"
system
  aaa
    admin-auth-order (local | radius | tacacs)
    auth-fallback
    auth-order (local | radius | tacacs)
    logs
      audit-disable
      netconf-disable
    radius-servers tag
    user user-name
      group group-name
      password password
    usergroup group-name
      task (interface | policy | routing | security | system) (read | write)
  admin-tech-on-failure
  archive
    interval minutes
    path file-path/filename
    ssh-id-file file-path/filename
    vpn vpn-id
  clock
    timezone timezone
  console-baud-rate rate
  control-session-pps rate
  description text
  device-groups group-name
  domain-id domain-id
  eco-friendly-mode (on vEdge Cloud routers only)
  gps-location (latitude decimal-degrees | longitude decimal-degrees)
  host-name string
  host-policer-pps rate (on vEdge routers only)
  icmp-error-pps rate
  idle-timeout minutes
  iptables-enable
  location string
  logging
    disk
      enable
```

```
    file
      name filename
      rotate number
      size megabytes
    priority priority
  host
    name (name | ip-address)
    port udp-port-number
    priority priority
    rate-limit number interval seconds
multicast-buffer-percent percentage (on vEdge routers only)
ntp
  keys
    authentication key-id md5 md5-key
    trusted key-id
  server (dns-server-address | ipv4-address)
    key key-id
    prefer
    source-interface interface-name
    version number
    vpn vpn-id
organization-name string
port-hop
port-offset number
radius
  retransmit number
  server ip-address
    auth-port port-number
    priority number
    secret-key key
    source-interface interface-name
    tag tag
    vpn vpn-id
  timeout seconds
route-consistency-check (on vEdge routers only)
site-id site-id
sp-organization-name name (on vBond orchestrators and vSmart controllers only)
system-ip ip-address
system-tunnel-mtu bytes
tacacs
  authentication authentication-type
  server ip-address
    auth-port port-number
    priority number
    secret-key key
    source-interface interface-name
    vpn vpn-id
  timeout seconds
tcp-optimization-enabled
timer
  dns-cache-timeout minutes
track-default-gateway
track-interface-tag number (on vEdge routers only)
track-transport
tracker tracker-name
  endpoint-dns-name dns-name
  endpoint-ip ip-address
  interval seconds
  multiplier number
  threshold milliseconds
upgrade-confirm minutes
[no] usb-controller (on vEdge 1000 and vEdge 2000 routers only)
vbond (dns-name | ip-address) [local] [port number] [ztp-server]
```

System Monitoring Commands

Use the following commands to monitor system-wide parameters:

show aaa usergroup
show control local-properties
show logging
show ntp associations
show ntp peer
show orchestrator local-properties
show running-config system
show system status
show uptime
show users

Additional Information

System and SNMP Overview

Routing

Unicast Overlay Routing Overview

The overlay network is controlled by the Viptela Overlay Management Protocol (OMP), which is at the heart of Viptela overlay routing. This solution allows the building of scalable, dynamic, on-demand, and secure VPNs. The Viptela solution uses a centralized controller for easy orchestration, with full policy control that includes granular access control and a scalable secure data plane between all edge nodes.

The Viptela solution allows edge nodes to communicate directly over any type of transport network, whether public WAN, internet, metro Ethernet, MPLS, or anything else.

OMP Routing Protocol

The Viptela Overlay Management Protocol (OMP) is the protocol responsible for establishing and maintaining the Viptela control plane. It provides the following services:

- Orchestration of overlay network communication, including connectivity among network sites, service chaining, and VPN topologies
- Distribution of service-level routing information and related location mappings
- Distribution of data plane security parameters
- Central control and distribution of routing policy

OMP is the control protocol that is used to exchange routing, policy, and management information between the vSmart controllers and vEdge routers in the overlay network. It is enabled by default, so after you start up the vSmart controllers and vEdge routers, it is not necessary to explicitly configure or enable OMP. These devices automatically initiate OMP peering sessions between themselves, and the two IP end points of the OMP session are the system IP addresses of the two devices.

OMP is an all-encompassing information management and distribution protocol that enables the overlay network by separating services from transport. Services provided in a typical VPN setting are usually located within a VPN domain, and they are protected so that they are not visible outside the VPN. In such a traditional architecture, it is a challenge to extend VPN domains and service connectivity.

OMP addresses these scalability challenges by providing an efficient way to manage service traffic based on the location of logical transport end points. This method extends the data plane and control plane separation concept from within routers to across the network. OMP distributes control plane information, along with related policies. A central vSmart controller makes all decisions related to routing and access policies for the overlay routing domain. OMP is then used to propagate routing, security, services, and policies that are used by edge devices for data plane connectivity and transport.

OMP Route Advertisements

On vSmart controllers and vEdge routers, OMP advertises to its peers the routes and services that it has learned from its local site, along with their corresponding transport location mappings, which are called TLOCs. These routes are called OMP routes or vRoutes, to distinguish them from standard IP routes. It is through OMP routes that the vSmart controllers learn the topology of the overlay network and the services available in the network.

OMP interacts with traditional routing at local sites in the overlay network. It imports information from traditional routing protocols, such as OSPF and BGP, and this routing information provides reachability within the local site. The importing of routing information from traditional routing protocols is subject to user-defined policies.

Because OMP operates in an overlay networking environment, the notion of routing peers is different from a traditional network environment. From a logical point of view, the overlay environment consists of a centralized controller and a number of edge devices. Each edge device advertises its imported routes to the centralized controller, and, based on policy decisions, this controller distributes the overlay routing information to other edge devices in the network. Edge devices never advertise routing information to each other, either using OMP or any other method. The OMP peering sessions between the centralized controller and the edge devices are used exclusively to exchange control plane traffic; they are never, in any situation, used for data traffic.

Registered edge devices automatically collect routes from directly connected networks, as well as static routes and routes learned from IGP protocols. The edge devices can also be configured to collect routes learned from BGP.

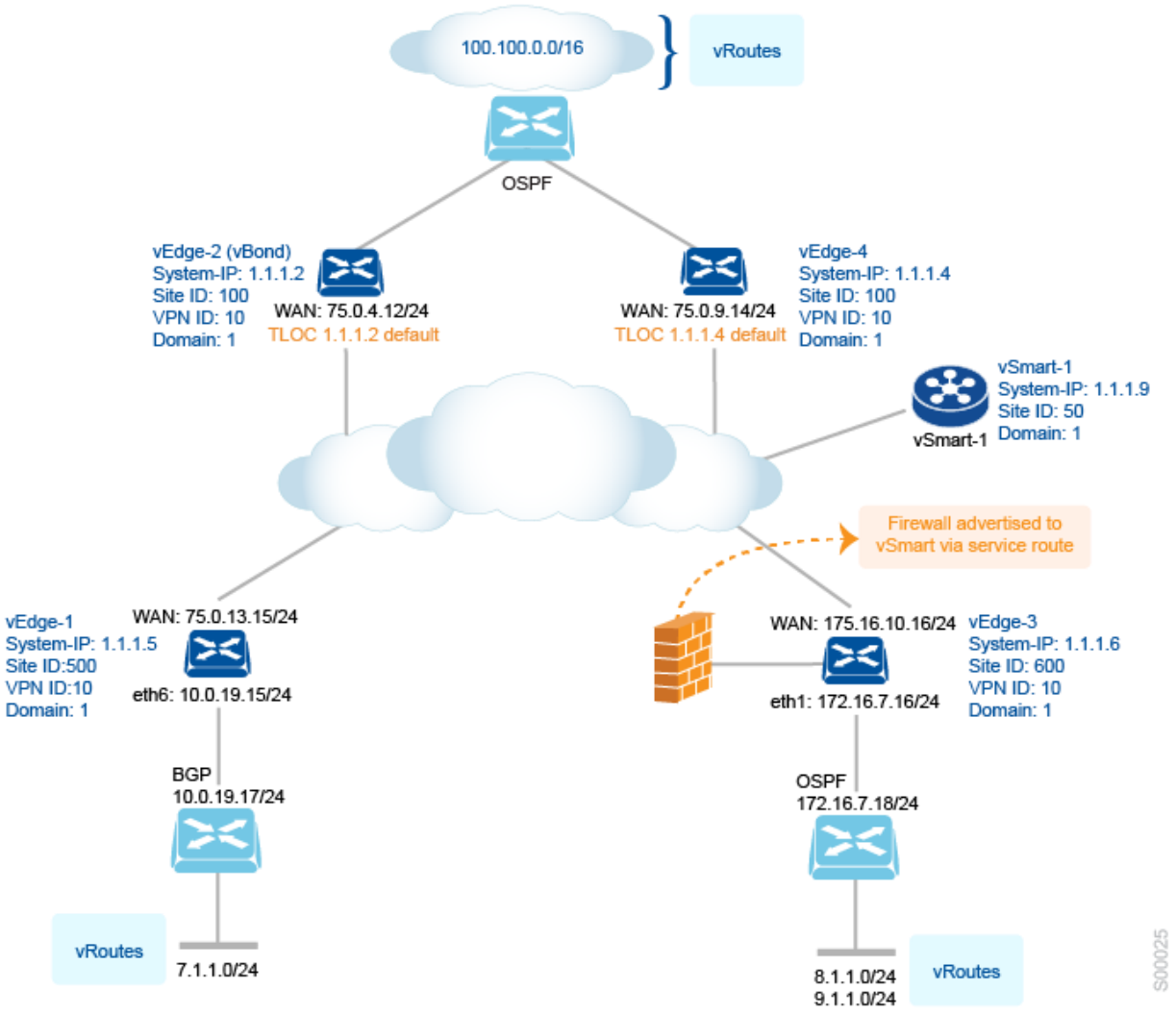
OMP performs path selection, loop avoidance, and policy implementation on each local device to decide which routes are installed in the local routing table of any edge device.

OMP advertises the following types of routes:

- OMP routes (also called vRoutes)—Prefixes that establish reachability between end points that use the OMP-orchestrated transport network. OMP routes can represent services in a central data center, services at a branch office, or collections of hosts and other end points in any location of the overlay network. OMP routes require and resolve into TLOCs for functional forwarding. In comparison with BGP, an OMP route is the equivalent of a prefix carried in any of the BGP AFI/SAFI fields.
- Transport locations (TLOCs)—Identifiers that tie an OMP route to a physical location. The TLOC is the only entity of the OMP routing domain that is visible to the underlying network, and it must be reachable via routing in the underlying network. A TLOC can be directly reachable via an entry in the routing table of the physical network, or it must be represented by a prefix residing on the outside of a NAT device and must be included in the routing table. In comparison with BGP, the TLOC acts as the next hop for OMP routes.
- Service routes—Identifiers that tie an OMP route to a service in the network, specifying the location of the service in the network. Services include firewalls, Intrusion Detection Systems (IDPs), and load balancers. Service route information is carried in both service and OMP routes.

(OMP also advertises policies configured on the vSmart controller that are executed on vEdge routers, including application-routing policy, cflowd flow templates, and data policy. For more information, see Policy Overview .)

The following figure illustrates the three types of OMP routes.



OMP Routes

Each vEdge router at a branch or local site advertises OMP routes to the vSmart controllers in its domain. These routes contain routing information that the vEdge router has learned from its site-local network.

A vEdge router can advertise one of the following types of site-local routes:

- Connected (also known as direct)
- Static
- BGP
- OSPF (inter-area, intra-area, and external)

OMP routes advertise the following attributes:

- TLOC—Transport location identifier of the next hop for the vRoute. It is similar to the BGP NEXT_HOP attribute. A TLOC consists of three components:

  - System IP address of the OMP speaker that originates the OMP route
  - Color to identify the link type
  - Encapsulation type on the transport tunnel

- Origin—Source of the route, such as BGP, OSPF, connected, and static, and the metric associated with the original route.
- Originator—OMP identifier of the originator of the route, which is the IP address from which the route was learned.
- Preference—Degree of preference for an OMP route. A higher preference value is more preferred.
- Service—Network service associated with the OMP route.
- Site ID—Identifier of a site within the Viptela overlay network domain to which the OMP route belongs.
- Tag—Optional, transitive path attribute that an OMP speaker can use to control the routing information it accepts, prefers, or redistributes.
- VPN—VPN or network segment to which the OMP route belongs.

You configure some of the OMP route attribute values, including the system IP, color, encapsulation type, carrier, preference, service, site ID, and VPN You can modify some of the OMP route attributes by provisioning control policy on the vSmart controller. See Centralized Control Policy .

TLOC Routes

TLOC routes identify transport locations. These are locations in the overlay network that connect to physical transport, such as the point at which a WAN interface connects to a carrier. A TLOC is denoted by a 3-tuple that consists of the system IP address of the OMP speaker, a color, and an encapsulation type. OMP advertises each TLOC separately.

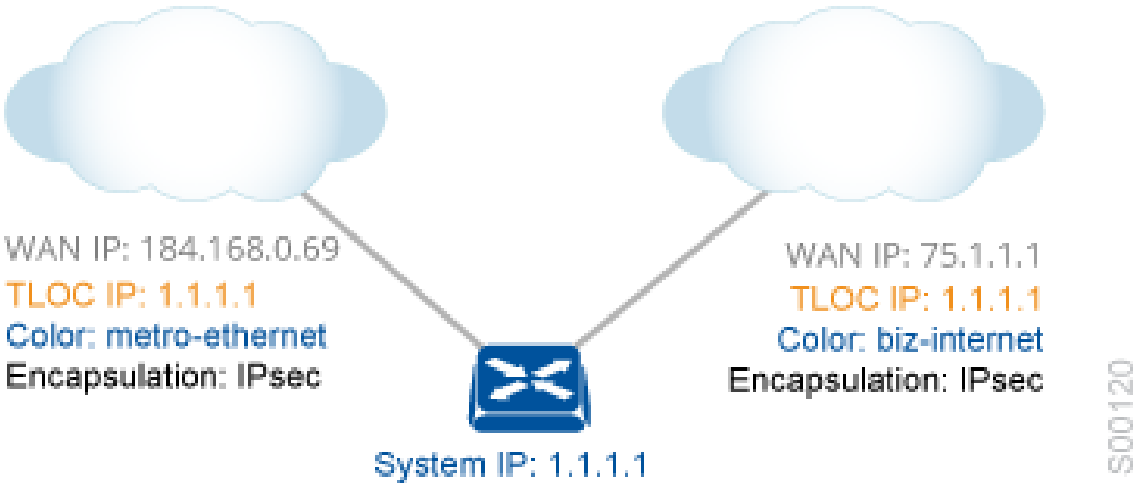
TLOC routes advertise the following attributes:

- TLOC private address—Private IP address of the interface associated with the TLOC.
- TLOC public address—NAT-translated address of the TLOC.
- Carrier—An identifier of the carrier type, which is generally used to indicate whether the transport is public or private.
- Color—Identifies the link type.
- Encapsulation type—Tunnel encapsulation type.
- Preference—Degree of preference that is used to differentiate between TLOCs that advertise the same OMP route.
- Site ID—Identifier of a site within the Viptela overlay network domain to which the TLOC belongs.
- Tag—Optional, transitive path attribute that an OMP speaker can use to control the flow of routing information toward a TLOC. When an OMP route is advertised along with its TLOC, both or either can be distributed with a community TAG, to be used to decide how send traffic to or receive traffic from a group of TLOCs.
- Weight—Value that is used to discriminate among multiple entry points if an OMP route is reachable through two or more TLOCs.

The IP address used in the TLOC is the fixed system address of the vEdge router itself. The reason for not using an IP address or an interface IP address to denote a TLOC is that IP addresses can move or change; for example, they can be assigned by DHCP, or interface cards can be swapped. Using the system IP address to identify a TLOC ensures that a transport end point can always be identified regardless of IP addressing.

The link color represents the type of WAN interfaces on vEdge router. The Viptela solution offers predefined colors, which are assigned in the configuration of the vEdge routers. The color can be one of default, 3g, biz-internet, blue, bronze, custom1, custom2, custom3, gold, green, lte, metro-ethernet, mpls, private1, private2, public-internet, red, and silver.

The encapsulation is that used on the tunnel interface. It can be either IPsec or GRE.



The diagram to the right shows a vEdge router that has two WAN connections and hence two TLOCs. The system IP address of the router is 1.1.1.1. The TLOC on the left is uniquely identified by the system IP address 1.1.1.1, the color metro-ethernet, and the encapsulation IPsec, and it maps to the physical WAN interface with the IP address 184.168.0.69. The TLOC on the right is uniquely identified by the system IP address 1.1.1.1, the color biz-internet, and the encapsulation IPsec, and it maps to the WAN IP address 75.1.1.1.

You configure some of the TLOC attributes, including the system IP address, color, and encapsulation, and you can modify some of them by provisioning control policy on the vSmart controller. See Centralized Control Policy .

Service Routes

Service routes represent services that are connected to a vEdge router or to the local-site network in which the vEdge router resides. The vEdge router advertises these routes to vSmart controllers using service address family NLRI. See Service Chaining .

OMP Route Redistribution

OMP automatically redistributes the following types of routes that it learns either locally or from its routing peers:

- Connected
- Static
- OSPF intra-area routes
- OSPF inter-area routes

To avoid routing loops and less than optimal routing, redistribution of following types of routes requires explicit configuration:

- BGP
- OSPF external routes

To avoid propagating excessive routing information from the edge to the access portion of the network, the routes that vEdge routers receive via OMP are not automatically redistributed into the other routing protocols running on the routers. If you want to redistribute the routes received via OMP, you must enable this redistribution locally, on each vEdge router.

OMP sets the origin and sub-origin type in each OMP route to indicate the route's origin (see the table below). When selecting routes, the vSmart controller and the vEdge routers take the origin type and subtype into consideration.

OMP Route Origin Type	OMP Route Origin Subtype
BGP	External Internal
Connected	—
OSPF	External-1 External-2 Intra-area Inter-area
Static	—

OMP also carries the metric of the original route. A metric of 0 indicates a connected route.

Administrative Distance

Administrative distance is the measure used to select the best path when there are two or more different routes to the same destination from multiple routing protocols. When the vSmart controller or vEdge router is selecting the OMP route to a destination, it prefers the one with the lower or lowest administrative distance value.

The following table lists the default administrative distances used by the Viptela devices:

Protocol	Administrative Distance
Connected	0
Static	1
NAT (NAT and static routes cannot coexist in the same VPN; NAT overwrites static routes)	1
Learned from DHCP	1
GRE	5
EBGP	20
OSPF	110
IBGP	200
OMP	250

OMP Best-Path Algorithm and Loop Avoidance

vEdge routers advertise their local routes to the vSmart controller using OMP. Depending on the network topology, some routes might be advertised from multiple vEdge routers. Viptela devices use the following algorithm to choose the best route:

1. Check whether the OMP route is valid. If not, ignore it.
2. If the OMP route is valid and if it has been learned from the same Viptela device, select the OMP route with the lower administrative distance.
3. If the administrative distances are equal, select the OMP route with the higher OMP route preference value.
4. On vEdge routers only, if the OMP route preference values are equal, select the OMP route with the higher TLOC preference value.

5. If the TLOC preference values are equal, compare the origin type, and select one in the following order (select the first match):
   Connected
   Static
   EBGP
   OSFP intra-area
   OSPF inter-area
   OSPF external
   IBGP
   Unknown

6. If the origin type is the same, select the OMP route that has the lower origin metric.

7. On vEdge routers only, if the origin types are the same, select the OMP route the higher router ID.

8. If the router IDs are equal, a vEdge router selects the OMP route with the higher private IP address. If a vSmart controller receives the same prefix from two different sites and if all attributes are equal, the vSmart controller chooses both of them.

Here are some examples of choosing the best route:

- A vSmart controller receives a OMP route to 10.10.10.0/24 via OMP from a vEdge router with an origin code of OSPF, and it also receives the same route from another vSmart controller, also with an origin code of OSPF. If all other things are equal, the best-path algorithm chooses the route that came from the vEdge router.

- A vSmart controller learns the same OMP route, 10.10.10.0/24, from two vEdge routers in the same site. If all other parameters are the same, both routes are chosen and advertised to other OMP peers. By default, up to four equal-cost routes are selected and advertised.

A vEdge router installs an OMP route in its forwarding table (FIB) only if the TLOC to which it points is active. For a TLOC to be active, an active BFD session must be associated with that TLOC. BFD sessions are established by each vEdge router, which creates a separate BFD session with each of the remote TLOCs. If a BFD session becomes inactive, the vSmart controller removes from the forwarding table all the OMP routes that point to that TLOC.

A vSmart controller runs the best path algorithm on received paths and stores the best paths in the winning order. It then walks through this list and applies outbound policy on each path. If a path is not rejected by the outbound policy, it is advertised to its peers. A max of configured send-path-limit paths for each prefix are advertised to the peers.

Here are some examples of choosing the paths to advertise to peer:

- A vSmart controller receives 8 paths for prefix 99.99.99.0/24 and the send-path-limit is set to 4. When the outbound policy does not reject any paths the first 4 paths from the received best paths stored in the winning order is selected.

- A vSmart controller receives 8 paths for prefix 99.99.99.0/24 and the send-path-limit is set to 4. When the outbound policy rejects some of the paths, the first 4 non-rejected paths from the received best paths stored in the winning order is selected.

Graceful Restart for OMP

Graceful restart for OMP allows the data plane in the Viptela overlay network to continue functioning if the control plane stops functioning or becomes unavailable. With graceful restart, if the vSmart controller in the network goes down, or if multiple vSmart controllers go down simultaneously, the vEdge routers can continue forwarding data traffic. They do this using the last known good information that they received from the vSmart controller. When a vSmart controller is again available, its DTLS connection to the vEdge router is re-established, and the vEdge router then receives updated, current network information from the vSmart controller.

Support for Traditional Unicast Routing Protocols

The Viptela overlay network supports BGP and OSPF unicast routing protocols. These protocols can be configured on vEdge routers in any VPN except for VPN 0 and VPN 512 to provide reachability to networks at their local sites. vEdge routers can redistribute route information learned from BGP and OSPF into OMP so that OMP can better choose paths within the overlay network.

When the local site connects to a Layer 3 VPN MPLS WAN cloud, the vEdge router acts as an MPLS CE device and establishes a BGP peering session to connect to the PE router in the L3VPN MPLS cloud.

When the vEdge router or routers at a local site do not connect directly to the WAN cloud but are one or more hops from the WAN and connect indirectly through a non-Viptela hub router, standard routing must be enabled on the vEdge routers' DTLS connections so that they can reach the WAN cloud. Either OSPF or BGP can be the routing protocol.

In both these types of topology, the BGP or OSPF session runs over a DTLS connection created on the loopback interface in VPN 0, which is the tranport VPN that is responsible for carrying control traffic in the overlay network. The vBond orchestrator learns about this DTLS connection via the loopback interface and conveys this information to the vSmart controller so that it can track the TLOC-related information. In VPN 0, you also configure the physical interface that connects the vEdge router to its neighbor—either the PE router in the MPLS case or the hub or next-hop router in the local site—but you do not establish a DTLS tunnel connection on that physical interface.

Additional Information

Configuring OMP

By default, OMP is enabled on all vEdge routers and vSmart controllers. OMP must be operational for the Viptela overlay network to function. If you disable it, you disable the overlay network.

This article describes how to provision OMP parameters.

Configure OMP Graceful Restart

OMP graceful restart allows OMP peers to continue operating if one of the peers becomes unavailable for some reason. If a vSmart controller becomes unavailable, its peer vEdge router continues to forward traffic, using the last-known good routing information received from the vSmart controller. Similarly, if a vEdge router becomes unavailable, its peer vSmart controller continues to use the last-known good routing information that it received from that vEdge router.

OMP graceful restart is enabled by default on vSmart controllers and vEdge routers. The default graceful restart time is 43,200 seconds (12 hours).

When OMP graceful restart is enabled, a vEdge router and a vSmart controller (that is, two OMP peers) cache the OMP information that they learn from their peer. This information includes OMP routes, TLOC routes, service routes, IPsec SA parameters, and centralized data policies. When one of the OMP peers is no longer available, the other peer uses the cached information to continue operating in the network. So, for example, when a vEdge router no longer detects the presence of the OMP connection to a vSmart controller, the router continues forwarding data traffic using the cached OMP information. The router also periodically checks whether the vSmart controller has again become available. When it does come back up and the vEdge router re-establishes a connection to it, the router flushes its local cache and considers only the new OMP information from the vSmart controller to be valid and reliable. This same scenario occurs when a vSmart controller no longer detects the presence of a vEdge router.

OMP graceful restart has a timer that tells the OMP peer how long to retain the cached advertised routes. When this timer expires, the cached routes are considered to be no longer valid, and the OMP peer flushes them from its route table. The default timer is 43,200 seconds (12 hours), and the timer range is 1 through 604,800 seconds (7 days). To modify the default timer value:

```
Viptela(config-omp)# timers graceful-restart-timer seconds
```

The graceful restart timer is set up independently on each OMP peer; that is, it is set up separately on each vEdge router and vSmart controller. To illustrate what this means, let's consider a vSmart controller that uses a graceful restart time of 300 seconds, or 5 minutes, and a vEdge router that is configured with a timer of 600 seconds (10 minutes). Here, the vSmart controller retains the OMP routes learned from that router for 10 minutes—the graceful restart timer value that is configured on the router and that the router has sent to the vSmart controller during the setup of the OMP session. The vEdge router retains the routes it learns from the vSmart controller for 5 minutes, which is the default graceful restart time value that is used on the vSmart controller and that the controller sent to the router, also during the setup of the OMP session.

While a vSmart controller is down and a vEdge router is using cached OMP information, if you reboot the vEdge router, it loses its cached information and hence will not be able to forward data traffic until it is able to establish a control plane connection to the vSmart controller.

To disable OMP graceful restart:

```
Viptela(config-omp)# no omp graceful-restart
```

Advertise Routes to OMP

By default, a vEdge router advertises connected, static routes, and OSPF inter-area and intra-area routes to OMP, and hence to the vSmart controller responsible for the vEdge router's domain. The router does not advertise BGP or OSPF external routes to OMP.

To have the vEdge router advertise these routes to OMP, and hence to the vSmart controller responsible for the vEdge router's domain, use the advertise command:

To configure the routes that the vEdge router advertises to OMP for all VPNs configured on the router:

```
vEdge(config-omp)# advertise (bgp | connected | ospf type | static)
```

To configure the routes that the vEdge router advertises to OMP for a specific VPN on the router:

```
vEdge(config-vpn-omp)# advertise (aggregate prefix [aggregate-only] | bgp | connected | network prefix |
ospf type | static)
```

For OSPF, the route type can be **external** .

The **bgp** , **connected** , **ospf** , and **static** options advertise all learned or configured routes of that type to OMP. To advertise a specific route instead of advertising all routes for a protocol, use the **network** option, specific the prefix of the route to advertise.

For individual VPNs, you can aggregate routes from the specified prefix before advertising them into OMP. By default, the aggregated prefixes and all individual prefixes are advertised. To advertise only the aggregated prefix, include the **aggregate-only** option.

Route advertisements that you set with the **omp advertise** command apply to all VPNs configured on the router. Route advertisements that you set with the **vpn omp advertise** command apply only to the specific VPN. If you configure route advertisements with both commands, they are both applied.

By default, when BGP advertises routes into OMP, BGP advertises each prefix's metric. BGP can also advertise the prefix's AS path:

```
vEdge(config)# vpn vpn-id router bgp
vEdge(config-bgp)# propagate-aspath
```

When you configure BGP to propagate AS path information, the router sends AS path information to routers that are behind the vEdge router (in the service-side network) that are running BGP, and it receives AS path information from these routers. If you are redistributing BGP routes into OMP, the AS path information is included in the advertised BGP routes. If you configure BGP AS path propagation on some but not all vEdge routers in the overlay network, the routers on which it is not configured receive the AS path information but they do not forward it to the BGP routers in their local service-side network. Propagating AS path information can help to avoid BGP routing loops.

In networks that have both overlay and underlay connectivity—for example, when vEdge routers are interconnected by both a Viptela overlay network and an MPLS underlay network—you can assign as AS number to OMP itself. For vEdge routers running BGP, this overlay AS number is included in the AS path of BGP route updates. To configure the overlay AS:

```
vEdge(config)# omp
vEdge(omp)# overlay-as as-number
```

You can specify the AS number in 2-byte ASDOT notation (1 through 65535) or in 4-byte ASDOT notation (1.0 through 65535.65535). As a best practice, it is recommended that the overlay AS number be a unique AS number within both the overlay and the underlay networks. That use, select an AS number that is not used elsewhere in the network.

If you configure the same overlay AS number on multiple vEdge routers in the overlay network, all these routers are considered to be part of the same AS, and as a result, they do not forward any routes that contain the overlay AS number. This mechanism is an additional technique for preventing BGP routing loops in the network.

Configure the Number of Advertised Routes

vEdge routers advertise the routes that they learn from their local site to the vSmart controller, and the vSmart controller redistributes this routes to other vEdge routers in the overlay network. The routes advertised are actually a tuple consisting of the route and the TLOC associated with that route.

A vEdge router can have up to six WAN interfaces, and each WAN interface has a different TLOC. (A WAN interface is any interface in VPN 0 that is configured as a tunnel interface. Both physical and loopback interfaces can be configured to be tunnel interfaces.) The vEdge router advertises each route–TLOC tuple to the vSmart controller.

The vSmart controller redistributes the routes it learns from vEdge routers, advertising each route–TLOC tuple. If, for example, a local site as two vEdge routers, a vSmart controller could potentially learn eight route–TLOC tuples for the same route.

By default, vEdge routers and vSmart controllers advertises up to four equal-cost route–TLOC tuples for the same route. You can configure them to advertise from 1 to 16 route–TLOC tuples for the same route:

```
Viptela(config-omp)# send-path-limit number
```

If the limit is lower than the number of route–TLOC tuples, the vEdge router or vSmart controller advertises the best routes.

Configure the Number of Installed OMP Paths

vEdge routers install OMP paths that they received from the vSmart controller into their local route table. By default, a vEdge router installs a maximum of four unique OMP paths into its route table. You can modify this number:

```
vEdge(config-omp)# ecmp-limit number
```

The maximum number of OMP paths installed can range from 1 through 16.

Configure the OMP Hold Time

The OMP hold time determines how long to wait before closing the OMP connection to a peer. If the peer does not receive three consecutive keepalive messages within the hold time, the OMP connection to the peer is closed. The default OMP hold time is 60 seconds. To modify the OMP hold time interval:

```
Viptela(config-omp)# timers holdtime seconds
```

The hold time can be in the range 0 through 65535 seconds.

The keepalive timer is one-third the hold time and is not configurable.

If the local device and the peer have different hold time intervals, the higher value is used.

If you set the hold time to 0, the keepalive and hold timers on the local device and the peer are set to 0.

The hold time must be at least two times the hello tolerance interval set on the WAN tunnel interface in VPN 0. To configure the hello tolerance interface, use the **hello-tolerance** command.

Configure the OMP Update Advertisement Interval

By default, OMP sends Update packets once per second. To modify this interval:

```
Viptela(config-omp)# timers advertisement-interval seconds
```

The interval can be in the range 0 through 65535 seconds.

Configure the End-of-RIB Timer

After an OMP session goes down and then comes back up, an end-of-RIB (EOR) marker is sent after 300 seconds (5 minutes). After this maker is sent, any routes that were not refreshed after the OMP session came back up are considered to be stale and are deleted from the route table. To modify the EOR timer:

```
Viptela(config-omp)# timers eor-timer seconds
```

The time can be in the range 1 through 3600 seconds (1 hour).

Additional Information

show omp peers
show omp routes
show omp services
show omp summary
show omp tlocs
Unicast Overlay Routing Overview

Configuring Unicast Overlay Routing

This article describes how to provision unicast overlay routing and the OSPF and BGP routing protocols.

Configure Service-Side Routing

The Viptela overlay network supports BGP and OSPF unicast routing protocols. You can configure these protocols on vEdge routers to provide reachability to networks at their local sites. Provisioning BGP and OSPF in this way enables routing on the service side of the network.

To set up routing on the vEdge router, you provision one VPN or multiple VPNs if segmentation is required. Within each VPN, you configure the interfaces that participate in that VPN and the routing protocols that operate in that VPN.

Because vSmart controllers never participate in a local site network, you never configure BGP or OSPF on these devices.

Set Up Basic OSPF on a vEdge Router

To configure basic service-side OSPF functionality:

1. Configure a VPN for the OSPF network:
   vEdge(config)# **vpn** *vpn-id*
   vpn-id can be any VPN number except VPN 0 and VPN512. VPN 0 is the transport VPN and carries only control traffic, and VPN 512 is the management interface.

2. Configure OSPF area 0 and the interfaces that participate in that area:
   vEdge(config-vpn)# **router ospf**
   vEdge(config-ospf)# **area 0**
   vEdge(config-area-0)# **interface** *interface-name*
   vEdge(config-interface)# **ip-address** *address*
   vEdge(config-interface)# **no shutdown**
   vEdge (ospf-if)# **exit**

3. Redistribute OMP routes into OSPF:
   vEdge(config-ospf)# **redistribute omp**
   By default, OMP routes are not redistributed into OSPF.

4. Repeat Steps 1 through 3 for any additional VPNs.

5. If desired, configure OMP to advertise to the vSmart controller any BGP and OSPF external routes that the vEdge router has learned:
   vEdge(config)# **omp**
   vEdge(config-omp)# **advertise bgp**
   vEdge(config-omp)# **advertise ospf external**

Here is an example of an OSPF routing configuration on the vEdge router. This configuration sets up VPN 10 with two interfaces, **ge2/0** and **ge3/0** . It enables OSPF routing on those interfaces in area 0, and it redistributes the OMP routes from the vSmart controller into OSPF.

```
vpn 10
  router
    ospf
      redistribute omp
      area 0
        interface ge2/0
        exit
      interface ge3/0
       exit
     exit
    !
  !
  interface ge2/0
    ip address 10.0.5.12/24
    no shutdown
  !
  interface ge3/0
    ip address 10.0.2.12/24
    no shutdown
  !
```

Set Up Basic BGP on a vEdge Router

To configure basic service-side BGP functionality:

1. Configure a VPN:
   vEdge(config)# **vpn** *vpn-id*
   vpn-id can be any service-side VPN, which is a VPN other than VPN 0 and VPN 512. VPN 0 is the transport VPN and carries only control traffic, and VPN 512 is the management VPN.

2. Configure BGP to run in the VPN:

   a. Configure the local AS number:
      vEdge(config-vpn)# **router bgp** *local-as-number*
      You can specify the AS number in 2-byte ASDOT notation (1 through 65535) or in 4-byte ASDOT notation (1.0 through 65535.65535).

   b. Configure the BGP peer, specifying its address and AS number (the remote AS number), and enable the connection to the peer:
      vEdge(config-bgp)# **neighbor** *address* **remote-as** *remote-as-number*
      vEdge(config-bgp)# **no shutdown**

3. Configure a system IP address for the vEdge router:
   vEdge(config)# **system system-ip** *address*

Here is an example of a BGP configuration on the vEdge router:

```
vEdge# show running-config system
system
  system-ip 10.1.2.3
!
vEdge# show running-config vpn 1
```

```
vpn 1
  router
    bgp 1
      neighbor 11.1.2.3
        no shutdown
        remote-as 2
      !
    !
  !
  ip route 0.0.0.0/0 10.0.16.13
!
```

Redistribute BGP Routes and AS Path Information

By default, routes from other routing protocols are not redistributed into BGP. It can be useful for BGP to learn OMP routes, because OMP learns routes to destinations throughout the overlay network. BGP on the vEdge router then advertises the OMP routes to all the BGP routers in the service-side of the network. To redistribute OMP routes into BGP so that these routes are advertised to all BGP routers in the service side of the network, configure redistribution in any VPN except VPN 0 or VPN 512:

```
vEdge(config)# vpn vpn-id router bgp
vEdge(config-bgp)# address-family ipv4-unicast redistribute omp [route-policy policy-name]
```

You can also redistribute routes learned from other protocols into BGP:

```
vEdge(config-bgp)# address-family ipv4-unicast redistribute (connected | nat | natpool-outside | ospf |
static) [route-policy policy-name]
```

You can control redistribution of routes on a per-neighbor basis:

```
vEdge(config-bgp)# neighbor ip-address
vEdge(config-neighbor)# address-family ipv4-unicast redistribute (connected | nat | natpool-outside | omp
| ospf | static)
vEdge(config-neighbor)# route-policy policy-name (in | out)
```

In the BGP route redistribution commands, the optional route policy is applied to the routes that are redistributed into BGP or routes that are redistributed out from BGP.

You can configure the vEdge router to advertise BGP routes that it has learned, through OMP, from the vSmart controller. Doing so allows the vSmart controller to advertise these routes to other vEdge routers in the overlay network. You can advertise BGP routes either globally or for a specific VPN:

```
vEdge(config)# omp advertise bgp

vEdge(config)# vpn vpn-id omp advertise bgp
```

By default, when BGP advertises routes into OMP, BGP advertises each prefix's metric. BGP can also advertise the prefix's AS path:

```
vEdge(config)# vpn vpn-id router bgp
vEdge(config-bgp)# propagate-aspath
```

When you configure BGP to propagate AS path information, the router sends AS path information to routers that are behind the vEdge router (in the service-side network) that are running BGP, and it receives AS path information from these routers. If you are redistributing BGP routes into OMP or into another protocol, or if you are advertising BGP routes to OMP, the AS path information is included in the advertised BGP routes. If you configure BGP AS path propagation on some but not all vEdge routers in the overlay network, the routers on which it is not configured receive the AS path information but they do not forward it to the BGP routers in their local service-side network. Propagating AS path information can help to avoid BGP routing loops.

In networks that have both overlay and underlay connectivity—for example, when vEdge routers are interconnected by both a Viptela overlay network and an MPLS underlay network—you can assign as AS number to OMP itself. For vEdge routers running BGP, this overlay AS number is included in the AS path of BGP route updates. To configure the overlay AS:

```
vEdge(config)# omp
vEdge(omp)# overlay-as as-number
```

You can specify the AS number in 2-byte ASDOT notation (1 through 65535) or in 4-byte ASDOT notation (1.0 through 65535.65535). As a best practice, it is recommended that the overlay AS number be a unique AS number within both the overlay and the underlay networks. That use, select an AS number that is not used elsewhere in the network.

If you configure the same overlay AS number on multiple vEdge routers in the overlay network, all these routers are considered to be part of the same AS, and as a result, they do not forward any routes that contain the overlay AS number. This mechanism is an additional technique for preventing BGP routing loops in the network.

Configure Transport-Side Routing

When a vEdge router is not directly connected to the WAN cloud, it cannot communicate with the vSmart controller in the overlay network. To enable communication between the vEdge router and other Viptela devices, you configure OSPF or BGP on a loopback interface in VPN 0. The loopback interface is a virtual transport interface that is the terminus of the DTLS and IPsec tunnel connections required for the vEdge router to participate in the overlay network.

To configure transport-side routing, you configure a loopback interface, the physical interface, and the routing protocol in VPN 0.

Configure BGP Transport-Side Routing

To configure BGP transport-side routing:

1. Configure a physical interface in VPN 0:
   vEdge(config)# **vpn 0 interface ge** *slot* **/** *port* **ip address** *address*
   vedge(config-interface)# **no shutdown**

2. Configure a loopback interface in VPN 0:
   vEdge(config)# **vpn 0 interface loopback** *number* **ip address** *address*
   vEdge(config-interface)# **no shutdown**
   vEdge(config-interface)# **tunnel-interface color** *color*

3. Configure a BGP instance in VPN 0:
   vEdge(config)# **vpn 0 router bgp** *local-as-number*

4. Create a policy for BGP to advertise the loopback interface address to its neighbors:
   vEdge(config)# **policy lists prefix-list** *prefix-list-name* **ip-prefix** *prefix*
   prefix is the IP address of the loopback interface.

5. Configure a route policy that affects the loopback interface's prefix:
   vEdge(config)# **policy route-policy** *policy-name* **sequence** *number* **match address** *prefix-list-name*
   vEdge(config)# **policy route-policy** *policy-name* **sequence** *number* **action accept**
   vEdge(config)# **policy route-policy** *policy-name* **default-action reject**

6. Reference the policy in the BGP instance. To apply the policy such that the loopback address is advertised to all BGP neighbors:
   vEdge(config)# **vpn 0 router bgp** *local-as-number* **address-family ipv4-unicast redistribute connected route-policy** *policy-name*

   To apply the policy only to a specific neighbor:
   vEdge(config)# **vpn 0 router bgp** *local-as-number* **neighbor** *neighbor-address* **address-family ipv4-unicast redistribute connected route-policy** *policy-name* **out**
   Specify **out** in the second command so that BGP advertises the loopback prefix out to the neighbor.

Here is an example of a minimal BGP transport-side routing configuration in which the loopback address is advertised to all the vEdge router's BGP neighbors. Note that even though we did not configure any services on the tunnel interface, these services are associated with the tunnel by default and are included in the configuration. Because services affect only physical interfaces, you can ignore them on loopback interfaces.

```
vEdge# show running-config vpn 0
vpn 0
 router
  bgp 2
   router-id 172.16.255.18
   timers
    keepalive 1
    holdtime  3
   !
   address-family ipv4-unicast
    redistribute connected route-policy export_loopback
   !
   neighbor 10.20.25.16
    no shutdown
    remote-as 1
    timers
     connect-retry          2
     advertisement-interval 1
    !
   !
  !
 !
 interface ge0/1
  ip address 10.20.25.18/24
  no shutdown
 !
 interface loopback
  ip address 172.16.255.118/32
  tunnel-interface
   color lte
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
!
policy
 lists
  prefix-list loopback_prefix
   ip-prefix 172.16.255.118/32
  !
 !
 route-policy export_loopback
  sequence 10
   match
    address loopback_prefix
   !
   action accept
   !
  !
  default-action reject
 !
!
```

Configure OSPF Transport-Side Routing

To configure OSPF transport-side routing:

1. Configure a physical interface in VPN 0:
   vEdge(config)# **vpn 0 interface ge** *slot* / *port* **ip address** *address*
   vEdge(config-interface)# **no shutdown**

2. Configure a loopback interface in VPN 0 as a tunnel interface:
   vEdge(config)# **vpn 0 interface loopback** *number* **ip address** *address*
   vEdge(config-interface)# **no shutdown**
   vEdge(config-interface)# **tunnel-interface color** *color*

3. Configure an OSPF instance in VPN 0:
   vEdge(config)# **vpn 0 router ospf**

4. Add the physical and loopback interfaces to the OSPF area:
   vEdge(config-ospf)# **area** *number* **interface ge** *slot* **/** *port*
   vEdge(config-area)# **interface loopback** *number*

Here is any example of a minimal OSPF transport-side routing configuration. Note that even though we did not configure any services on the tunnel interface, these services are associated with the tunnel by default and are included in the configuration. Because services affect only physical interfaces, you can ignore them on loopback interfaces.

```
vEdge# show running-config vpn 0
vpn 0
 router
  ospf
   router-id 172.16.255.11
   timers spf 200 1000 10000
   area 0
    interface ge0/1
    exit
    interface loopback1
    exit
   exit
  !
 !
 interface ge0/1
  ip address 10.0.26.11/24
  no shutdown
 !
 interface loopback1
  ip address 10.0.101.1/32
  tunnel-interface
   color lte
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
!
```

Additional Information

Unicast Overlay Routing Overview
Routing Configuration Example
show bgp neighbor
show ip mfib summary
show ip routes
show ospf neighbor

Multicast Overlay Routing Overview

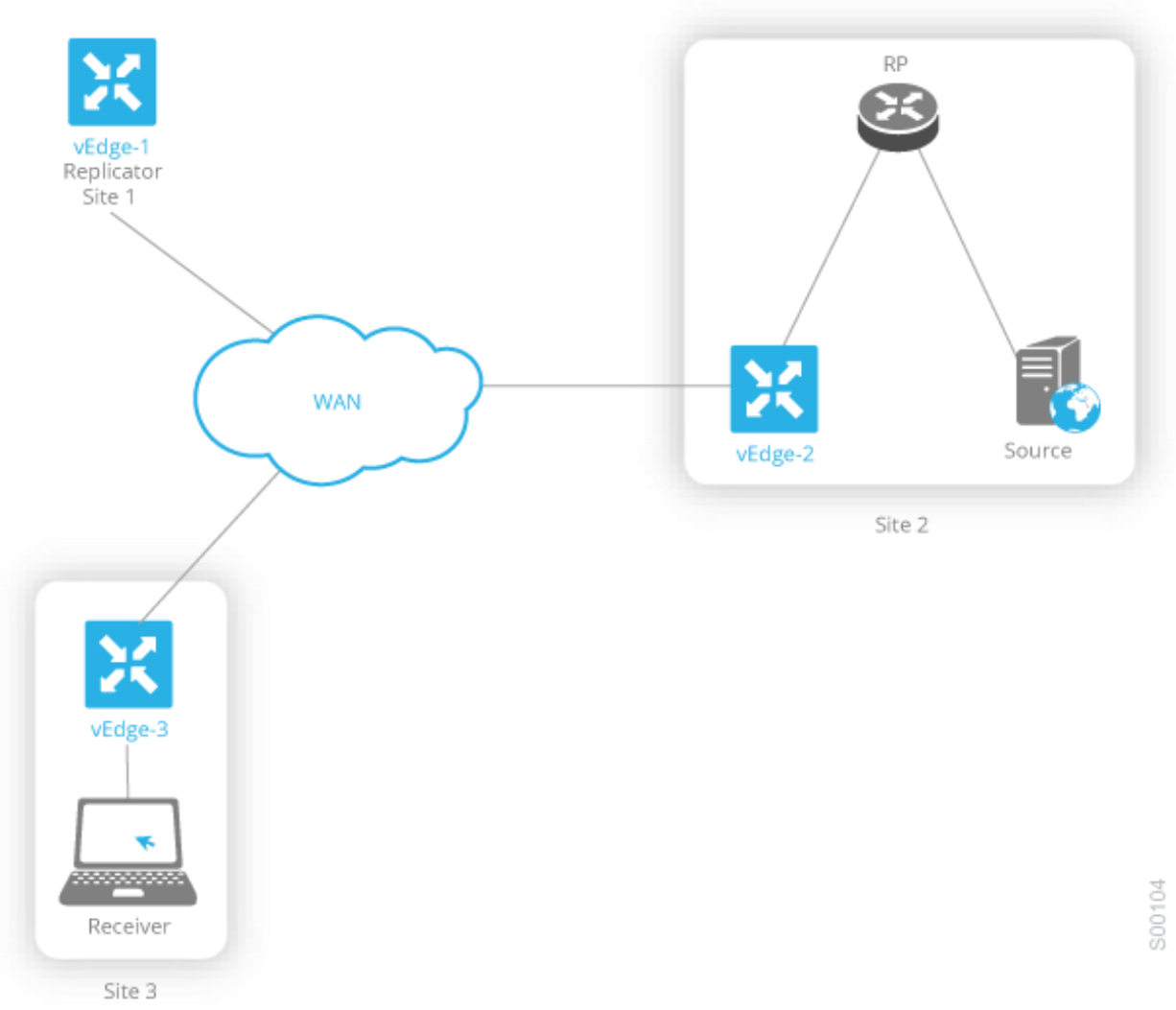
The Viptela overlay multicast implementation extends native multicast by creating a secure optimized multicast tree that runs on top of the overlay network.

Overlay Multicast Routing Overview

The Viptela overlay multicast software uses PIM Sparse Mode (PIM-SM) for multicasting traffic on the overlay network. PIM-SM builds unidirectional shared trees rooted at a rendezvous point (RP), and each multicast group has one shared tree that is rooted at a single RP. Once a shared tree has been built such that a last-hop router learns the IP address for the multicast source, the router engages in a switchover from the shared tree to initiate the construction of a source (or shortest-path) tree. The source tree uses the lowest metric path between the source and last-hop router, which may be entirely, partially, or not at all congruent with the shared tree.

The Viptela design optimizes multicast packet distribution throughout the overlay network by eliminating packet replication on the ingress router, that is, on the router connected to a multicast source. Instead, the ingress router forwards multicast streams to a vEdge router that is designated to be a replicator, and it is this router that forwards streams to multicast receivers. This design saves bandwidth and computational resources on the ingress router.

The figure below illustrates the Viptela design. Here, the ingress router vEdge-2 forwards the multicast streams from the source at its site to vEdge-1, which is a vEdge router that is designated to be a replicator. vEdge-1 replicates the stream and forwards it to the receiver, which is located behind vEdge-3 at Site 3.



Multicast Protocols

VIptela overlay multicast supports the PIM routing protocol and IGMP.

PIM

Viptela overlay multicast supports PIM version 2 (defined in RFC 4601), with some restrictions.

On the service side, the Viptela software supports native multicast. A vEdge router appears as a native PIM router and establishes PIM neighborship with other PIM routers at a local site. To properly extend multicast trees into the overlay network, a vEdge router may require other supporting routers in a local site. If a PIM-SM RP is required at a site, that function must be provided by a non-Viptela router, because the vEdge router currently has no native support for the rendervouz point functionality. Receivers residing downstream of a vEdge router can join multicast streams by exchanging IGMP membership reports directly with the device, and no other routers are required. This applies only to sites that have no requirement for supporting local sources or PIM SM rendezvouz points.

On the transport side, PIM-enabled vEdge routers originate multicast service routes (called multicast autodiscover routes),sending them via OMP to the vSmart controllers. The multicast autodiscover routes indicate whether the router has PIM enabled and whether it is a

replicator. If the router is a replicator and the load threshold has been configured, this information is also included in the multicast autodiscover routes. Each PIM router also conveys information learned from the PIM join messages sent by local-site multicast-enabled routers, including multicast group state, source information, and RPs. These routes assist vEdge routers in performing optimized joins across the overlay when joining existing multicast sources.

vEdge routers support PIM source-specific mode (SSM), which allows a multicast source to be directly connected to the router.

IGMP

Viptela overlay multicast routing supports the Internet Group Management Protocol (IGMP) version 2 (defined in RFC 2236). vEdge routers use IGMP to process receiver membership reports for the hosts in a particular VPN and to determine, for a given group, whether multicast traffic should be forwarded and state should be maintained. vEdge routers listen for both IGMPv1 and IGMPv2 group membership reports.

Rendezvous Points

The root of a PIM multicast shared tree resides on a router configured to be a rendezvous point (RP). Each RP acts as the RP and the root of a shared tree (or trees) for specific multicast group ranges. In the Viptela overlay network, RPs are non-Viptela routers that reside in the local-site network. The RP function is typically assigned to one or two locations in the network; it is not required at every site. vEdge routers do not currently support the RP functionality, so non-Viptela routers must provide this function in the applicable sites.

The Viptela software supports the auto-RP protocol for distributing RP-to-group mapping information to local-site PIM routers. With this information, each PIM router has the ability to forward joins to the correct RP for the group that a downstream IGMP client is attempting to join. Auto-RP updates are propagated to downstream PIM routers if such routers are present in the local site.

Replicators

For efficient use of WAN bandwidth, strategic vEdge routers can be deployed and configured as replicators throughout the overlay network. Replicators mitigate the requirement for an ingress router to replicate a multicast stream once for each receiver.

As discussed above, replicators advertise themselves, via OMP multicast-autodiscover routes, to the vSmart controllers in the overlay network. The controllers then forward the replicator location information to the PIM-enabled vEdge routers that are in the same VPN as the replicator.

A replicator vEdge router receives streams from multicast sources, replicates them, and forwards them to multicast receivers. The details of the replication process are discussed below, in the section Multicast Traffic Flow through the Overlay Network.

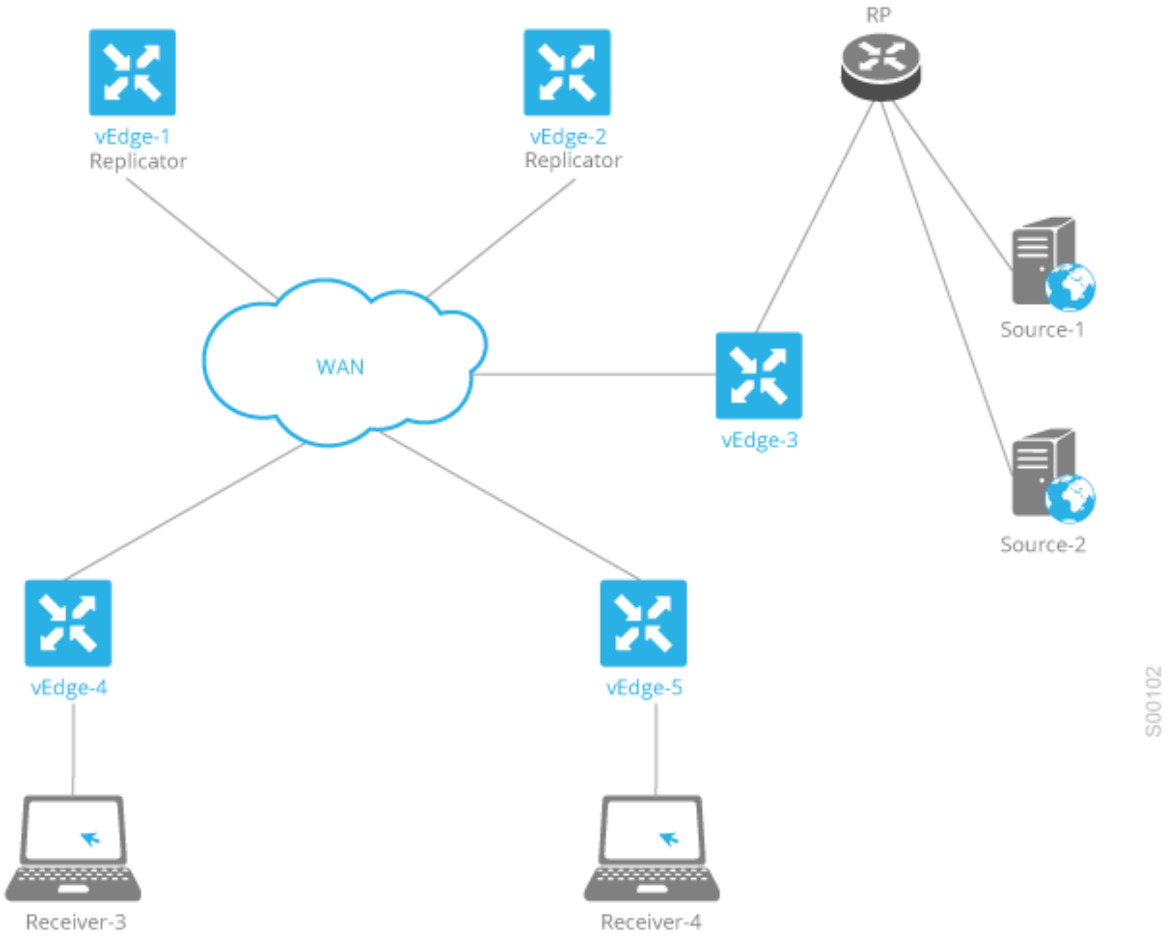
A replicator is typically vEdge router located at a colo site or another site with a higher-speed, or a high-speed, connection to the WAN transport network.

Multicast Service Routes

vEdge routers send multicast service routes to the vSmart controller via OMP. From these routes, the controller processes and forwards joins for requested multicast groups towards the source address as specified in the original PIM join message that helped originate the OMP multicast service route. The source address can be either the IP address of an RP if the originating router is attempting to join the shared tree or the IP address of the actual source of the multicast stream if the originating router is attempting to join the source tree.

Multicast Traffic Flow through the Overlay Network

Let’s look at a high-level topology of the Viptela overlay network multicast solution to illustrate how traffic from multicast sources is delivered to multicast receivers. The topology contains five vEdge routers:

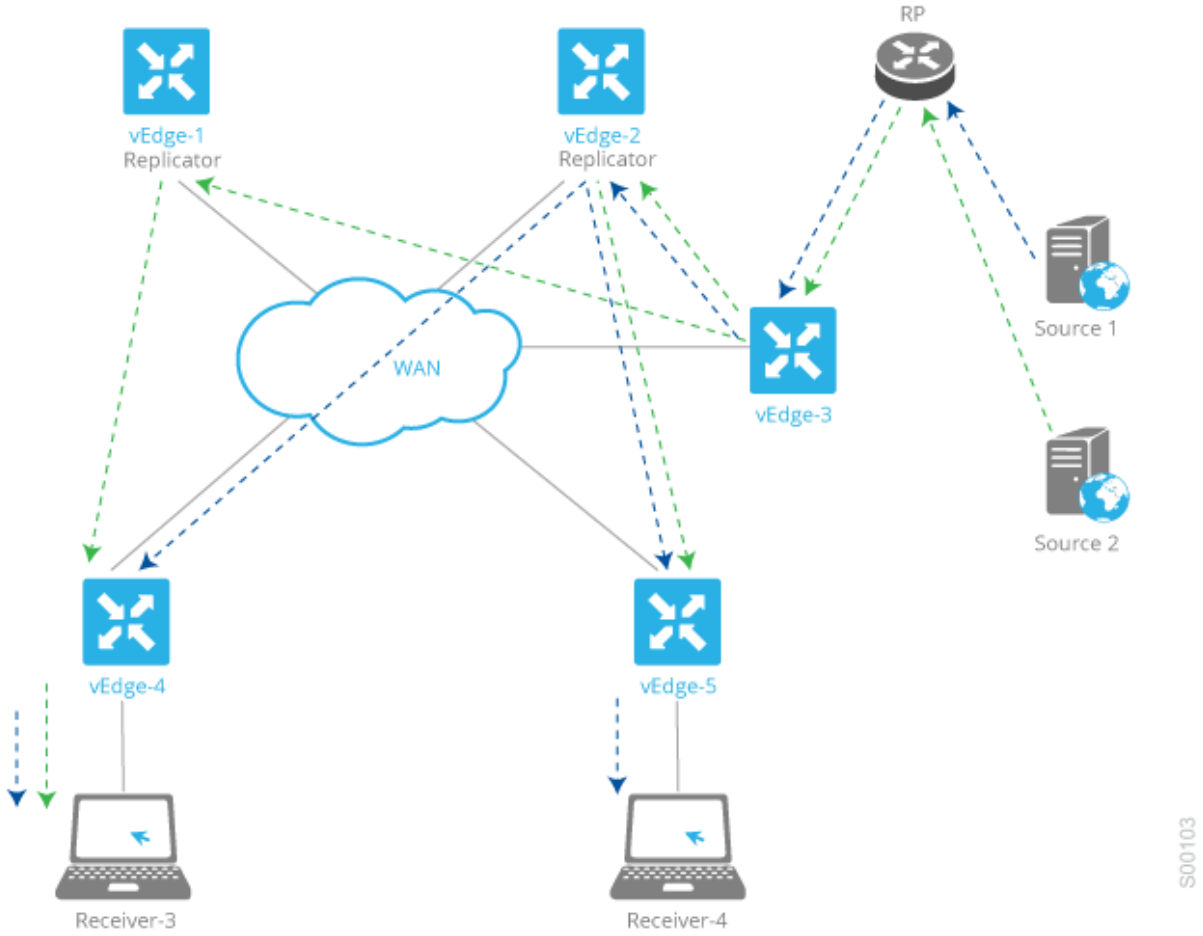


- vEdge router vEdge-3 is located at a site with two multicast sources, Source-1 and Source-2. This site also has a non-Viptela router that functions as a PIM-SM RP. Even though the vEdge-3 router is the ingress router for streams from these two multicast sources, it performs no packet replication. Instead, it forwards the multicast streams to replicators in the overlay network. The vEdge-3 router has learned the addresses of the replicators via OMP from a vSmart controller.
- vEdge routers vEdge-1 and vEdge-2 are two multicast replicators in the overlay network. Their job as replicators is to receive streams from multicast sources, replicate the streams, and then forward them to receivers. In this topology, the vEdge-3 router forwards the multicast streams from the two multicast sources in its local network to vEdge-1 or vEdge2, or both, and these routers then replicate and forward the streams to the receivers located in the local sites behind vEdge routers vEdge-4 and vEdge5. Which replicator receives a stream depends on the group address, the identity of the vEdge routers that joins that given group, and the current load of the replicator. The typical situation is that only a single replicator is replicating traffic for a given group, but this may vary depending on the physical scope of the given group.
- vEdge router vEdge4 is located at a site that has one multicast receiver, Receiver-3, which receives streams from Source-1 and Source-2.
- vEdge router vEdge5 is located at another site with one multicast receiver, Receiver-4. This receiver gets streams only from one source, Source-1.

Now, let's examine how multicast traffic flows from the sources to the receivers.

The two multicast sources, Source-1 and Source-2, send their multicast streams (the blue stream from Source-1 and the green stream from Source-2) to the RP. Because the destination IP addresses for both streams are at remote sites, the RP forwards them to vEdge-3 for transmission onto the transport/WAN network. vEdge-3 has learned from the vSmart controller that the network has two replicators, vEdge-1 and vEdge-2, and so forwards the two multicast streams to them, without first replicating the streams.

The two replicators have learned from a vSmart controller the locations of multicast receivers for the two streams. The vEdge-1 replicator makes one copy of the green stream and forwards it to vEdge-4, which in turns forwards it to the Receiver-3. The vEdge-2 replicator makes one copy of the green stream, which it forwards to vEdge-5 (from which it goes on to Receiver-4), and it makes two copies of the blue stream, which it forwarda to vEdge-4 and vEdge-5 (and which they then forward to the two receivers).



Now, let's look at the multicast configurations on the five vEdge routers:

- vEdge router vEdge-1 is a PIM replicator for a particular VPN. If we assume that no multicast sources, receivers, or RPs are located in its local network, the configuration of this router is simple: In the VPN, enable the replicator functionality, with the **router multicast-replicator local** command, and enable PIM, with the **router pim** command.
- vEdge router vEdge-2 also acts only as a replicator in the same VPN as vEdge-1, and you configure it with the same commands, **router multicast-replicator local** and **router pim** , when configuring the VPN. Each replicator can accept a maximum number of new PIM joins, and when this threshold value is reached, all new joins are sent to the second replicator. (If there is only one replicator, new joins exceeding the threshold are dropped.)
- vEdge router vEdge-4 runs PIM. You enable PIM explicitly on the service side within a VPN, specifying the service-side interface that connects to the multicast domain in the local network. So within the VPN, you include the **router pim interface** command. You can also enable auto-RP with the **router pim auto-rp** command.

  On the transport side, no explicit configuration is required. The vEdge router automatically directs multicast traffic—both OMP control plane messages and multicast streams—to VPN 0, which is the WAN transport VPN.
- vEdge router vEdge-5 is also configured to run PIM in the same way as vEdge-4: You configure the service-side interface name and RP information.

On all five of these vEdge routers, PIM must be enabled in the same VPN so that the multicast streams can be transmitted and received.

Additional Information

Configuring Multicast Overlay Routing

Configuring Multicast Overlay Routing

For any vEdge routers to be able to participate in the multicast overlay network, you configure PIM on those routers. You can optionally configure IGMP to allow individual hosts on the service side to join multicast groups within a particular VPN.

Enable PIM at a Site with Multicast Sources

For a vEdge router located at a site that contains one or more multicast sources, you enable PIM on the service-side interface or interfaces. These are the interfaces that face the local-site network. You enable PIM per VPN, so you must configure PIM and PIM interfaces for all VPNs support multicast services. You cannot configure PIM in VPN 0 (the transport VPN facing the overlay network) or in VPN 512 (the management VPN).

For each VPN, you must configure the name of the service-side interface. You can optionally configure auto-RP to receive group-to-RP mapping updates.

To configure PIM at a site with multicast sources:

1. Configure a VPN for the PIM network:
   vEdge(config)# **vpn** *vpn-id*
   vpn-id can be any VPN number except VPN 0 (the transport VPN facing the overlay network) or VPN 512 (the management VPN).

2. Configure the interfaces in the VPN:
   vEdge(config-vpn)# **interface** *interface-name*
   vEdge(config-interface)# **ip address** *prefix* **/** *length*
   vEdge(config-interface)# **no shutdown**
   The interface names in the two **interface** names must be the same.

3. Configure PIM and the interfaces that participate in the PIM network:
   vEdge(config-vpn)# **router pim**
   vEdge(config-pim)# **interface** *interface-name*
   vEdge(config-interface)# **no shutdown**
   The interface name in the two **interface** commands must be the same.

4. Optionally, modify PIM timers on the interface. The default PIM hello interval is 30 seconds, and the default join/prune interval is 60 seconds.
   vEdge(config-interface)# **hello-interval** *seconds*
   vEdge(config-interface)# **join-prune-interval** *seconds*
   The hello interval can be in the range of 1 through 3600 seconds. The join/prune interval can be in the range of 10 through 600 seconds.

5. Optionally, enable automatic discover of rendezvous points (RPs) in the PIM network:
   vEdge(config-pim)# **auto-rp**

Here is an example of a PIM configuration on a vEdge router:

```
vpn 10
 router
  pim
    interface ge1/1
      no shutdown
    auto-rp
```

Enable PIM at a Site with Multicast Receivers

For a vEdge router located at a site that contains one or more multicast receivers, you enable PIM on the service-side interface or interfaces (the interfaces facing the local-site network). You enable PIM per VPN, so you must configure PIM and PIM interfaces for all VPNs support multicast services.

For each VPN, you must configure the name of the service-side interface.

To configure PIM at a site with multicast receivers:

1. Configure a VPN for the PIM network:
   vEdge(config)# **vpn** *vpn-id*
   vpn-id can be any VPN number except VPN 0 (reserved for control plane traffic) or VPN 512 (the management VPN).

2. Configure PIM and the interfaces that participate in the PIM network:
   vEdge(config-vpn)# **router pim**
   vEdge(config-pim)# **interface** *interface-name*

3. Configure the interface used by PIM in the PIM VPN:
   vEdge(config-vpn)# **interface** *interface-name*
   vEdge(config-interface)# **ip address** *prefix* **/** *length*
   vEdge(config-interface)# **no shutdown**
   The interface names in the two **interface** names must be the same.

4. By default, a vEdge router joins the shortest-path tree (SPT) immediately after the first packet arrives from a new source. To force traffic to remain on the shared tree and travel via the RP instead of via the SPT, configure the traffic rate at which to switch from the shared tree to the SPT:
   vEdge(config-vpn)# **router pim** **spt-threshold** *kbps*
   The rate can be from 0 through 100 kbps.

5. In a topology that includes multicast replicators, the Viptela software, by default, uses the same replicator for a multicast group. You can have the software choose the replicator randomly:
   vEdge(config-vpn)# **router pim** **replicator-selection** **random**

Here is an example of a PIM configuration on a vEdge router:

```
vEdge(config-vpn-2)# show full-configuration
vpn 2
 router
  pim
   interface ge0/7
   exit
  exit
 !
 interface ge0/7
  ip address 10.0.100.15/24
  no shutdown
 !
!
```

Configure a Multicast Replicator

For a vEdge router that is a replicator, the configuration has two parts: you configure the router to be a replicator, and you enable PIM on each VPN that participates in a multicast domain.

To configure a replicator:

1. Configure a VPN for the PIM network:
   vEdge(config)# **vpn** *vpn-id*
   vpn-id can be any VPN number except VPN 0 (the transport VPN facing the overlay network) or VPN 512 (the management VPN).

2. Configure the replicator functionality on the local vEdge router:
   vEdge(config-vpn)# **router multicast-replicator local**

3. On the transport side, a single vEdge router acting as a replicator can accept a maximum of 1024 (*,G) and (S,G) joins. For each join, the router can accept 256 tunnel outgoing interfaces (OILs). To modify the number of joins the replicator can accept, change the value of the join threshold:
   vEdge(config-router)# **multicast-replicator threshold** *number*

4. Enable PIM on each VPN that participates in a multicast domain:
   vEdge(config)# **vpn** *vpn-id*
   vEdge(config-vpn)# **router pim**
   If the router is just a replicator and is not part of a local network that contains either multicast sources or receivers, you do not need to configure any interfaces in the PIM portion of the configuration. The replicator learns the locations of multicast sources and receivers from the OMP messages it exchanges with the vSmart controller. These control plane messages are exchanged in the transport VPN (VPN 0). Similarly, the other vEdge routers discover replicators dynamically, through OMP messages from the vSmart controller.

PIM Scalability Information

When configuring PIM, the following scalability limits apply:

- Any single vEdge router supports a maximum of 1024 multicast state entries. Note that a (*,G) and an (S,G) for the same group count as two entries.
- The 1024 multicast state entries are shared across all configured VPNs on a single vEdge router.
- Each state entry can contain a maximum of 64 service-side entries and a maximum of 256 transport-side entries in its outgoing interface list (OIL).

Enable IGMP at a Site with Multicast Hosts

For VPNs in which you want to individual hosts to join multicast groups, you can enable IGMP on vEdge routers:

```
vEdge(config)#vpn vpn-id router igmp
vEdge(config-igmp)# interface interface-name
```

Ensure that the interface being used for IGMP is configured in the VPN:

```
vEdge(config)# vpn vpn-id
vEdge(config-vpn)# interface interface-name
vEdge(config-interface)# ip address prefix/length
vEdge(config-interface)# no shutdown
```

If desired, specify the multicast groups to initiate join requests with:

```
vEdge(config-igmp)# interface interface-name join-group group-ip-address
```

Configure the Interface Bandwidth Allowed for Multicast Traffic

By default, multicast traffic can use up to 20 percent of the interface bandwidth. You can change this allocation to a value from 5 to 100 percent:

```
vEdge(config)# system multicast-buffer-percent percentage
```

This systemwide configuration applies to all multicast-enabled interfaces on the vEdge router.

Multicast Configuration Limitations

You cannot configure the following for overlay multicast routing:

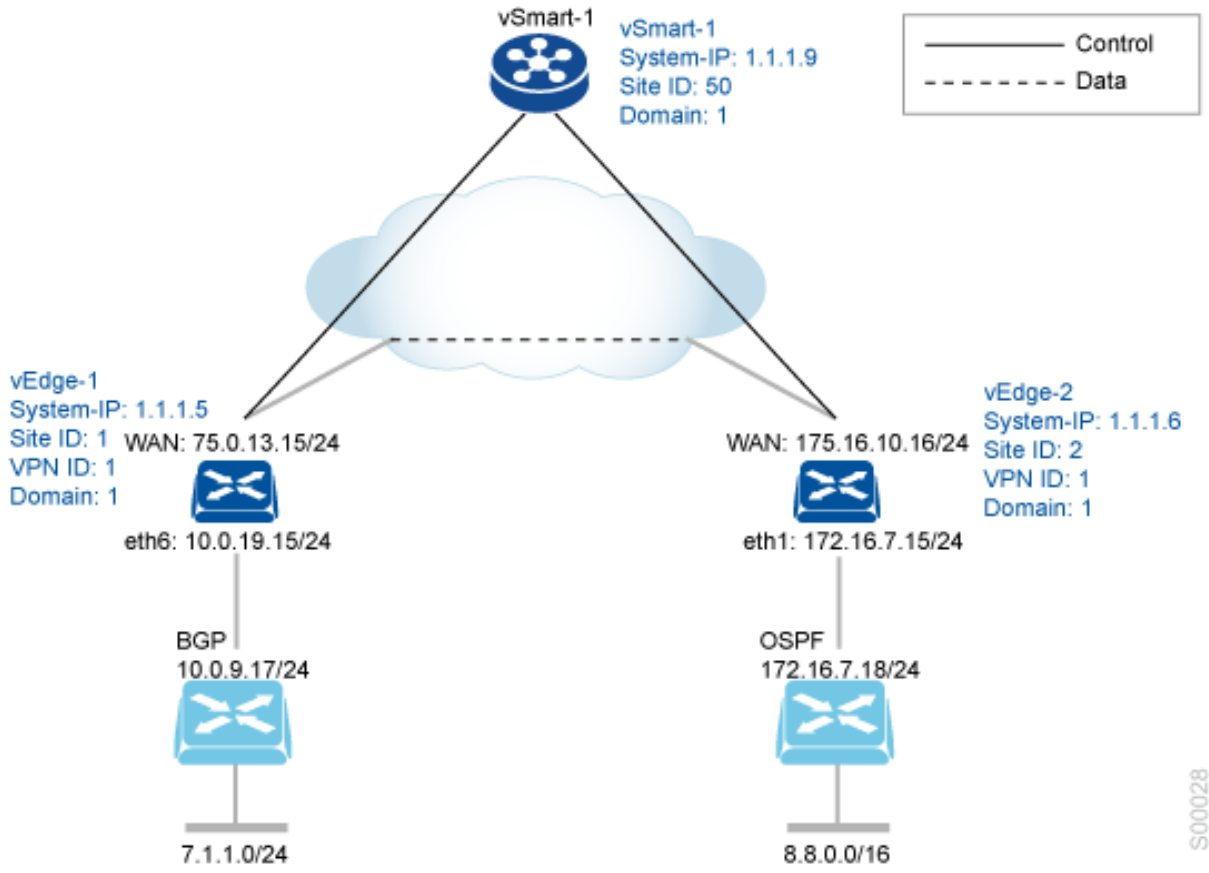
- Data policy
- Access lists
- Mirroring

Additional Information

Multicast Overlay Routing Overview

Routing Configuration Example

This example illustrates how to set up unicast routing in a Viptela overlay network. This network consists of one vBond orchestrator, one vSmart controller, and two vEdge routers at two different sites, as shown in the figure. The table following the figure shows the parameters for these devices.



	vBond Orchestrator	vSmart Controller	vEdge-1 Router	vEdge-2 Router
Public IP address	184.168.0.69	—	—	—
Domain	—	1	1	1

Site ID	—	50	1	2
VPN	—	Controller for VPN 1	1	1
System IP address	—	1.1.1.9	1.1.1.5	1.1.1.6
WAN-facing interface	—	Ethernet 4 (eth4)	ge1/1	ge2/1
IP address of WAN-facing interface	—	10.0.16.19/24	75.0.13.15/24	172.16.10.16/24
Branch-facing interface	—	—	ge0/1	ge0/2
IP address of branch-facing interface	—	—	10.0.19.15/24	172.16.7.15/24
IP address of vEdge-facing branch router	—	—	10.0.9.17/24	172.16.17.18/24
Branch address space	—	—	7.1.1.0/24	8.8.0.0/16
Branch routing protocol	—	—	BGP	OSPF
Branch AS number	—	—	—	1

vSmart Controller Configuration

On the vSmart controller:

1. Configure the system properties of the vSmart controller:
   vSmart1(config)# **system system-ip 1.1.1.9**
   vSmart1(config-system)# **domain-id 1**
   vSmart1(config-system)# **site-id 50**
   vSmart1(config-system)# **vbond 184.168.0.69**

2. Configure the physical transport information:

   a. Configure the IP address of the WAN-facing interface:
      vSmart1(config)# **vpn 0 interface eth4 ip address 10.0.16.19/24**

   b. Allow the interface to carry control-plane traffic:
      vSmart1(config)# **vpn 0 interface eth4 tunnel-interface**

   c. Enable the interface:
      vSmart1(config)# **vpn 0 interface eth4 no shutdown**

   d. Configure a default route to the WAN interface that connects to the transport cloud:
      vSmart1(config)# **vpn 0 ip route 0.0.0.0/0 10.0.16.19**

3. Commit the configuration:
   vSmart1(config)# **commit**

Here is the full configuration on the vSmart controller:

```
system
  system-ip 1.1.1.9
  domain-id 1
  site-id   50
  vbond 184.168.0.69
!
vpn 0
  interface eth4
    ip address 10.0.16.19/24
    tunnel-interface
```

```
      !
      no shutdown
    !
    ip route 0.0.0.0/0 10.0.16.19
!
```

vEdge1 Router Configuration

On the vEdge1 router:

1. OMP is enabled by default on the vEdge router. This branch network is running BGP, and we want to advertise BGP-learned routes to the vSmart controller:
   vEdge1(config)# **omp advertise bgp**

2. Configure the system properties of the vEdge1 router and the IP address of the vBond orchestrator:
   vEdge1(config)# **system system-ip 1.1.1.5**
   vEdge1(config-system)# **domain-id 1**
   vEdge1(config-system)# **site-id 1**
   vEdge1(config-system)# **vbond 184.168.0.69**

3. Configure the transport VPN and the transport interface:
   vEdge1(config)# **vpn 0 interface ge1/1 ip address 75.0.13.15/24**
   vEdge1(config-ge1-1)# **tunnel-interface**
   vEdge1(config-ge1-1)# **no shutdown** /span>

4. Configure the default route in the transport VPN:
   vEdge1(config-vpn-0)# **ip route 0.0.0.0/0 75.0.13.15**

5. For the VPN, configure BGP:

   a. Configure the local AS number:
      vEdge1(config)# **vpn 1 router bgp 1**

   b. Have BGP advertise only unicast traffic:
      vEdge1(config-bgp-1)# **address-family ipv4_unicast**

   c. Have BGP redistribute routes that it learns, via the OMP session, from the vSmart controller:
      vEdge1(config-bgp-1)# **redistribute omp**

   d. Enable and configure the BGP peer:
      vEdge1(config-bgp-1)# **neighbor 10.0.17.17 remote-as 2**
      vEdge1(config-bgp-1)# **no shutdown**

   e. Configure the interface between the vEdge1 router and its local-site router:
      vEdge1(config-bgp-1)# **vpn 1 interface ge0/1**
      vEdge1(config-interface-ge1-1)# **no shutdown**

6. Activate the configuration:
   vEdge1(config)# **commit**

Here is the full configuration on the vEdge1 router:

```
omp
  no shutdown
  advertise bgp
!
system
  system-ip 1.1.1.5
  domain-id 1
  site-id   1
```

```
  vbond 184.168.0.69
!
vpn 0
  interface ge1/1
    ip address 75.0.13.15/24
    tunnel-interface
    !
    no shutdown
  ip route 0.0.0.0/0 75.0.13.15
!
vpn 1
 router
   bgp 1
     address-family ipv4_unicast
     redistribute omp
     !
     neighbor 10.0.17.17
       no shutdown
       remote-as 2
     !
   !
 !
 interface ge0/1
   ip address 10.0.19.15/24
 !
!
```

vEdge2 Router Configuration

On the vEdge2 router:

1. OMP is enabled by default on the vEdge router. This branch network is running OSPF, and OSPF automatically redistributes its learned intra-area and inter-area routes to the vSmart controller. We also want to advertise OSPF external routes to the vSmart controller:
   vEdge2(config)# **omp advertise ospf external**

2. Configure the system properties of the vEdge2 router and the IP address of the vBond orchestrator:
   vEdge2(config)# **system system-ip 1.1.1.6**
   vEdge2(config-system)# **domain-id 1**
   vEdge2(config-system)# **site-id 2**
   vEdge2(config-system)# **vbond 184.168.0.69**

3. Configure the transport VPN and the transport interface and the default route:
   vEdge2(config)# **vpn 0 interface ge2/1 ip address 172.16.10.16/24**
   vEdge2(config-ge2-1)# **tunnel-interface**
   vEdge2(config-ge2-1)# **no shutdown**

4. Configure the default route in the transport VPN:
   vEdge2(config-vpn-0)# **ip route 0.0.0.0/0 172.16.10.16**

5. For the VPN, configure OSPF:

   a. Configure area 0 and add the ge0/2 interface to this area:
      vEdge2(config)# **vpn 2 router ospf area 0 interface ge0/2**

   b. Configure the interface between the vEdge2 router and its local-site router:
      vEdge2(config)# **vpn 2 interface ge0/2 ip address 172.16.7.16/24**
      vEdge2(config-interface-eth1)# **no shutdown**

6. Activate the configuration:
   vEdge1(config)# **commit**

Here is the full configuration on the vEdge2 router:

```
omp
  no shutdown
  advertise ospf external
!
system
 system-ip 1.1.1.6
 domain-id 1
 site-id   2
 vbond 184.168.0.69
!
vpn 0
  interface ge2/1
    ip address 172.16.10.16/24
    tunnel-interface
    !
    no shutdown
  !
  ip route 0.0.0.0/0 172.16.7.16
!
vpn 2
  router
    ospf
     area 0
        interface ge0/2
     exit
   exit
   !
!
 interface ge0/2
   ip address 172.16.7.16/24
   no shutdown
 !
!
```

Additional Information

Unicast Overlay Routing Overview
Configuring Unicast Overlay Routing
Multicast Overlay Routing Overview
Configuring Multicast Overlay Routing

Routing CLI Reference

CLI commands for configuring and monitoring the BGP, IGMP, OMP, OSPF, and PIM routing protocols on vEdge routers.

Routing Protocol Configuration Commands

You configure routing protocols on vEdge routers. For BGP and OSPF, you configure them in each VPN:

```
omp...
vpn vpn-id
  router
    bgp...
    igmp...
    multicast-replicator...
    ospf...
    pim...
```

BGP Configuration and Monitoring Commands

Use the following commands to configure BGP within a VPN on a vEdge router:

```
vpn vpn-id
  router
    bgp local-as-number
      address-family ipv4-unicast
        aggregate-address prefix/length [as-set] [summary-only]
        maximum-paths paths number
        network prefix/length
        redistribute (connected | nat | omp | ospf | static) [route-policy policy-name]
      best-path
        as-path multipath-relax
        compare-router-id
        med (always-compare | deterministic | missing-as-worst)
      distance
        external number
        internal number
        local number
      neighbor ip-address
        address-family ipv4-unicast
          maximum-prefixes number [threshold] [restart minutes | warning-only]
          route-policy policy-name (in | out)
        capability-negotiate
        description text
        ebgp-multihop ttl
        next-hop-self
        password md5-digest-string
        remote-as remote-as-number
        send-community
        send-ext-community
        [no] shutdown
        timers
          advertisement-interval number
          connect-retry seconds
          holdtime seconds
          keepalive seconds
        update-source ip-address
        ! end neighbor configuration
      propagate-aspath
      router-id ip-address
      [no] shutdown
      timers
        holdtime seconds
        keepalive seconds
```

Use the following commands to monitor BGP on a vEdge router:

- **clear bgp all** — Reset the connections to all BGP neighbors in the specified VPN.
- **clear bgp neighbor** —Reset the connection to a specific BGP neighbor in the specified VPN.
- **debug bgp events** —Display the events that have occurred as part of the BGP finite state model.
- **debug bgp fsm** —Display the states in the BGP finite state model, which describes the actions that BGP takes and the BGP packets that are exchanged between BGP neighbors when they are establishing a peering session.
- **debug bgp ipcs** —Display information about BGP interprocess communication with other processes running on the vEdge router.
- **debug bgp packets v** —Display the packets that BGP is receiving from its peers.
- **show bgp neighbor** —Display information about all BGP connections to neighbors.
- **show bgp routes** —Display information about all routes learned by BGP.

- **show bgp summary** —Display summary information about BGP.

IGMP Configuration and Monitoring Commands

Use the following commands to configure IGMP within a VPN on a vEdge router:

```
vpn vpn-id
  router
    igmp
      interface interface-name
        join-group group-address
      [no] shutdown
```

Use the following commands to monitor IGMP :

- **clear igmp interface** —Clear the interfaces on which IGMP is enabled.
- **clear igmp protocol** —Flush all IGMP groups and relearn them.
- **clear igmp statistics** —Zero IGMP statistics.
- **show igmp groups** —Display information about multicast groups.
- **show igmp interface** —Display information about the interfaces on which IGMP is enabled.
- **show igmp statistics** —Display IGMP statistics.

OMP Configuration and Monitoring Commands

By default, OMP is enabled on all vEdge routers and vSmart controllers. Use the following commands to modify the OMP configuration:

```
omp
  advertise (bgp | connected | ospf | static) (on vEdge routers only)
  discard-rejected (on vSmart controllers only)
  ecmp-limit number (on vEdge routers only)
  graceful-restart
  overlay-as as-number
  send-backup-paths (on vSmart controllers only)
  send-path-limit number
  [no] shutdown
  timers
    advertisement-interval seconds
    eor-timer seconds
    graceful-restart-timer seconds
    holdtime seconds
vpn
  omp
    advertise (aggregate prefix [aggregate-only] | bgp | connected | network prefix | ospf type | static)
(on vEdge routers only)
```

Use the following commands to monitor OMP:

- **clear omp all** —Restart OMP and all OMP sessions.
- **clear omp peer** —Restart the OMP session to a specific peer
- **clear omp routes** —Flush the OMP routes from the route table and then re-install them.
- **clear omp tlocs** —Remove TLOCs from the tunnel table and then re-install them.

- **debug omp events** [**level** ***level***]—Display information to help debug OMP events.
- **debug omp ipcs** [**level** ***level***]—Display information to help debug OSPF interprocess communication with other processes running on the Viptela device.
- **debug omp packets** [**level** ***level***]—Display information to help debug OMP packets.
- **show omp peers** —Display information about active OMP peering sessions.
- **show omp routes** —Display OMP route information.
- **show omp services** —Display OMP services information.
- **show omp summary** —Display summary information about OMP.
- **show omp tloc-paths** —Display TLOC path information.
- **show omp tlocs** —Display TLOC information.

OSPF Configuration and Monitoring Commands

Use the following commands to configure OSPF within a VPN on a vEdge router:

```
vpn vpn-id
  router
    ospf
      area number
        interface interface-name
          authentication
            authentication-key key
            message-digest key
            type (message-digest | simple)
          cost number
          dead-interval seconds
          hello-interval seconds
          network (broadcast | point-to-point)
          passive-interface
          priority number
          retransmit-interval seconds
        ! end area interface
        nssa
          no-summary
          translate (always | candidate | never)
        range prefix/length
          cost number
          no-advertise
        stub
          no-summary
    ! end area
    auto-cost reference-bandwidth mbps
    compatible rfc1583
    default-information
      originate (always | metric metric | metric-type type)
    distance
      external number
      inter-area number
      intra-area number
    max-metric
      router-lsa (administrative | on-startup seconds)
    redistribute (bgp | connected | nat | omp | static) route-policy policy-name
    route-policy policy-name in
    router-id ip-v4-address
```

```
    timers
      spf delay initial-hold-time maximum-hold-time
```

Use the following commands to monitor OSPF on a vEdge router:

- **clear ospf all** —Reset all OSPF neighbors in the specified VPN.
- **debug ospf events** —Display information to help debug OSPF events in the specified VPN.
- **debug ospf ipcs** —Display information to help debug OSPF interprocess communication with other processes running on the vEdge router.
- **debug ospf ism** —Display information about the OSPF interface state machine in the specified VPN, to help debug issues with establishing an OSPF session on an interface.
- **debug ospf lsa** —Display information to help debug OSPF link-state advertisements in the specified VPN.
- **debug ospf nsm** —Display information about the OSPF neighbor state machine in the specified VPN, to help debug issues with establishing an OSPF session with a neighbor.
- **debug ospf nssa** —Display information to help debug an OSPF NSSA (Not-So-Stubby Area)
- **debug ospf packets** —Display information about OSPF packets in the specified VPN,to help debug issues related to the exchange of packets between OSPF routers.
- **show ospf database** —Display the entries in the OSPF link-state database.
- **show ospf database-summary** —Display a summary of the entries in the OSPF link-state database.
- **show ospf interface** —List the interfaces in the specified VPN that are running OSPF.
- **show ospf neighbor** —List the OSPF neighbors in the specified VPN.
- **show ospf process** —List the OSPF processes running in the specified VPN.
- **show ospf routes** —Display information about all routes learned by OSPF.

PIM and Multicast Replicator Configuration and Monitoring Commands

Use the following commands to configure PIM and multicast replicators within a VPN on a vEdge router:

```
vpn vpn-id
  router
    multicast-replicator local [threshold number]
vpn vpn-id
  router
    pim
      auto-rp
      interface interface-name
        hello-interval seconds
        join-prune-interval seconds
      replicator-selection
      [no] shutdown
      spt-threshold kbps
```

Use the following commands to monitor PIM and multicast replicators :

- **clear ip mfib record** —Clear the statistics for a particular group, source, or VPN from the Multicast Forwarding Information Base (MFIB).
- **clear ip mfib stats** —Clear all statistics from the MFIB.
- **clear pim interface** —Relearn all PIM neighbors and joins.

- **clear pim neighbor** —Clear the statistics for a PIM neighbor.
- **clear pim protocol** —Clear all PIM protocol state.
- **clear pim statistics** —Clear all PIM-related statistcs and relearn all PIM neighbors and joins.
- **show ip mfib oil** —Display the list of outgoing interfaces from the MFIB.
- **show ip mfib stats** —Display packet transmission and receipt statistics for active entries in the MFIB.
- **show ip mfib summary** —Display a summary of all active entries in the MFIB.
- **show multicast replicator —** List information about multicast replicators.
- **show multicast rpf** —List multicast reverse-path forwarding information.
- **show multicast topology** —List information related to the multicast domain topology.
- **show multicast tunnel** —List information about the IPsec tunnels between multicast peers.
- **show omp multicast-auto-discover** —List the peers that support multicast.
- **show omp multicast-routes** —List the multicast routes that OMP has learned from PIM join messages.
- **show pim interface** —List the interfaces that are running PIM.
- **show pim neighbor** —List PIM neighbors.
- **show pim statistics** —Display all PIM-related statistics.

Additional Information

Configuring Multicast Overlay Routing
Configuring OMP
Configuring Unicast Overlay Routing
Multicast Overlay Routing Overview
Routing Configuration Examples
Unicast Overlay Routing Overview

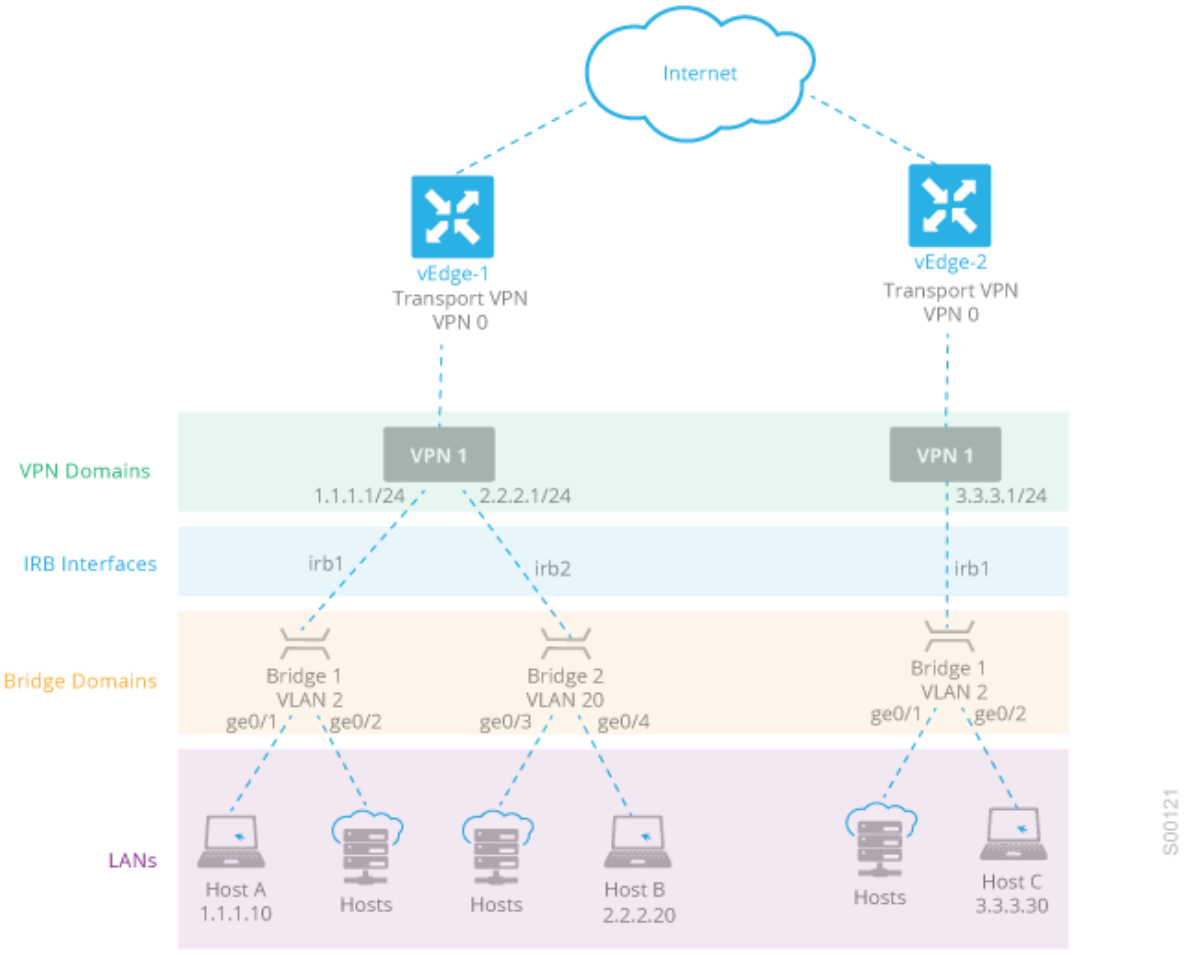
Bridging

Bridging Overview

A vEdge router can act as a transparent bridge, switching traffic between LANs that are part of a VLAN at the local router's site. To implement bridging, the Viptela architecture defines the concept of a *bridge domain* . Each bridge domain corresponds to a single VLAN. From a switching point of view, each bridge domain is a separate broadcast domain, and each has its own Ethernet switching table (or MAC table) to use for switching traffic within the broadcast domain. Multiple bridge domains, and hence multiple VLANs, can co-exist on a single vEdge router.

To allow hosts in different bridge domains to communicate with each other, vEdge routers support *integrated routing and bridging* (IRB). IRB is implemented using *logical IRB interfaces* , which connect a bridge domain to a VPN, or what might better be called a *VPN domain* . The VPN domain provides the Layer 3 routing services necessary so that traffic can be exchanged between different VLANs. Each bridge domain can have a single IRB interface and can connect to a single VPN domain, and a single VPN domain can connect to multiple bridge domains on a vEdge router. The route table in the VPN domain provides reachability between all bridge domains which participate in that VPN domain, whether the bridge domain is located on the local router or on a remote router.

The following figure illustrates the components of the Viptela bridging solution, and the remainder of this article describes these components.



Bridge Domains

In standard transparent bridging, virtual LANs, or VLANs, segregate LANs into logical LANs, and each VLAN is an isolated broadcast domain. All VLAN traffic remains in the VLAN, and it is directed to its destination by means of Ethernet switching tables. The Viptela implementation of bridging overlays the concept of a *bridge domain* on top of the standard VLAN: A bridge domain comprises a single VLAN, and all the ports within a VLAN are part of a single broadcast domain. Within each broadcast domain, the standard bridging operations of learning, forwarding, flooding, filtering, and aging are performed on VLAN traffic to create and maintain the Ethernet switching table (or MAC table) for that VLAN, and hence for that bridge domain.

Each bridge domain is identified by a number. The VLAN within a bridge domain is identified by an 802.1Q identifier, which is called a VLAN tag or VLAN ID. Frames within a bridge domain can remain untagged, or you can configure VLAN ID to tag the frames. In the Viptela design, the VLAN and the VLAN ID are the property of the bridge domain. They are not the property of an interface or a switching port.

Ports that connect to the WAN segments are associated with a bridge domain. In the Viptela overlay network, these ports are the physical Gigabit Ethernet interfaces on vEdge routers. Specifically, they are the base interfaces, for example, **ge-0/0** . You cannot use subinterfaces for bridge domain ports.

Each broadcast domain in the Viptela overlay network is uniquely identified by the combination of bridge domain number and VLAN ID (if configured). This design means that The same VLAN ID can be used in different bridge domains on a single vEdge router. For example, the VLAN ID 2 can exist in bridge domain 1 and bridge domain 50. In a situation where the VLAN IDs are different two bridge domains can include the same port interfaces. For example, both (bridge 2, VLAN 2) and (bridge 10, VLAN 23) can include interfaces ge0/0 and ge0/1. Here, these two interfaces effectively become trunk ports. However, because of how interface names are tracked internally, two bridge domains that use the same VLAN ID can have no overlap between the interfaces in the two domains. For example, if (bridge 1, VLAN 2) includes interfaces ge0/0 and ge0/1, these interfaces cannot be in (bridge 50, VLAN 2).

As mentioned above, all member interfaces within a VLAN are part of a single broadcast domain. Within each broadcast domain, the standard transparent bridging operations of learning, forwarding, flooding, filtering, and aging are performed on VLAN traffic to create and maintain the Ethernet switching table, also called the MAC table, for that VLAN.

The Viptela bridging domain architecture lacks the concepts of access ports and trunk ports. However, the Viptela architecure emulates these functions. For a vEdge router that has a single bridge domain, the interfaces in the bridge emulate access ports and so the router is similar to a single switch device. For a vEdge router with multiple bridge domains that are tagged with VLAN IDs, the interfaces in the bridges emulate trunk ports, and you can think of each domain as corresponding to a separate switching device.

Native VLAN

Viptela bridge domains support 802.1Q native VLAN. All traffic sent and received on an interface configured for native VLAN do not have a VLAN tag in its Ethernet frame. That is, they are not tagged with a VLAN ID. If a host is connected on an interface enabled for native VLAN, the bridge domain receives no tagged frames. If the bridge domain connects to a switch that support trunk ports or connects to a hub, the bridge domain might receive both untagged and tagged frames.

Native VLAN is used primarily on trunk ports.

Native VLAN provides backwards compatibility for devices that do not support VLAN tagging. For example, native VLAN allows trunk ports to accept all traffic regardless of what devices are connected to the port. Without native VLAN, the trunk ports would accept traffic only from devices that support VLAN tagging.

Integrated Routing and Bridging (IRB)

Bridge domains and VLANs provide a means to divide a LAN into smaller broadcast domains. Each VLAN is a separate broadcast domain, and switching within that domain directs traffic to destinations within the VLAN. The result is that hosts within a single bridge domain can communicate among themselves, but cannot communicate with hosts in other VLANs. So, for example, if a business places its departments in a separate VLANs, people within the finance department would be able to communicate only with others in that department, but would not be able to communicate with the manufacturing or engineering department.

The only way for traffic to cross Layer 2 VLAN boundaries to allow communicatation between bridge domains is via Layer 3 routing. This process of marrying switching and routing is done by *integrated routing and bridging* , or IRB. With IRB, a single vEdge router can pass

traffic among different bridge domains on the same router and among bridge domains on remote vEdge routers. The only restriction is that all the bridge domains must reside in the same VPN domain in the overlay network.

The Viptela implementation of IRB connects a Layer 2 bridge domain to a Layer 3 VPN domain via an IRB interface. An IRB interface is a logical interface that inherits all the properties of a regular interface, but it is not associated with a port or with a physical interface. Each IRB interface is named with the stem “irb” and a number that matches the number of a bridge domain. For example, the interface **irb2** is the logical interface that connects to bridge domain 2. IRB interfaces cannot have subinterfaces.

You create IRB interfaces within a VPN. A VPN domain supports multiple IRB interfaces.

There is a one-to-one association between an IRB logical interface and a bridge domain: an IRB interface can be associated only with one bridge domain, and a bridge domain can be associated with only one IRB interface. As a result, a bridge domain can be part of only one VPN in the overlay network.

The IP address of an IRB interface is the subnet of the VLAN that resides in the bridge domain. From a switching perspective, the IP address of the IRB interface is part of the bridge domain.

Additional Information

Configuring Bridging and IRB

Configuring Bridging and IRB

This article describes how to configuring Layer 2 bridging and Layer 3 integrated routing and bridging (IRB) on vEdge routers.

Configuring Bridging and Bridge Domains

Bridge domains can be marked with a VLAN tag, or they can remain untagged.

Create a Bridge Domain That Uses VLAN Tagging

For a bridge domain that uses VLAN tagging, a tag, called a VLAN ID, is inserted into all frame headers sent by the domain This tag identifies which VLAN the frames belong to, and it is used to determine which interfaces the vEdge router should send broadcast packets to.

To configure a bridge domain that uses VLAN tagging, create a bridging domain, assign a VLAN tag to that domain, and associate an interface with the domain:

1. Create a bridging domain:
   vEdge(config)# **bridge** *bridge-id*
   Each domain is identified by a unique integer, in the range 1 through 63. Each vEdge router can have up to 16 bridging domains.

2. Tag the bridging domain with a VLAN ID:
   vEdge(config-bridge)# **vlan** *number*
   The VLAN identifier can be a value from 1 through 4095.

3. Associate an interface with the bridging domain, and enable that interface:
   vEdge(config-bridge)# **interface ge** *slot* **/** *port*
   vEdge(config-interface)# **no shutdown**
   The interface must be a physical interface. You cannot use subinterfaces.

After you have added physical interfaces to a VLAN, if you want to change the VLAN identifier, you must first delete all the interfaces from the VLAN. Then configure a new VLAN identifier, and re-add the interfaces to the VLAN.

You can also configure these optional parameters:

1. Configure a description for the VLAN interface, to help identify the interface in operational command output:
   vEdge(config-bridge)# **interface ge** *slot* **/** *port*
   vEdge(config-interface)# **description "** *text description* **"**

2. Configure a static MAC address for the VLAN interface:
   vEdge(config-interface)# **static-mac-address** *aa* **:** *bb* **:** *cc* **:** *dd* **:** *ee* **:** ff

3. Configure a name for the VLAN, to help identify the VLAN in operational command output:
   vEdge(config-bridge)# **name** " *text description* "

4. By default, a bridging domain can learn up to 1024 MAC addresses. You can modify this to a value from 0 through 4096:
   vEdge(config-bridge)# **max-macs** *number*

5. By default, MAC table entries age out after 300 seconds (5 minutes). You can modify this to a value from 10 through 4096 seconds:
   vEdge(config-bridge)# **age-time** *seconds*

Here is an example configuration:

```
vEdge# config
vEdge(config)# bridge 2
vEdge(bridge-2)# vlan 27
vEdge(bridge-2)# interface ge0/4
vEdge(interface-ge0-4)# no shutdown
vEdge(interface-ge0-4)# description "VLAN tag = 27"
vEdge(interface-ge0/4)# commit and-quit
vEdge# show running-config bridge
bridge 2
 vlan 27
 interface ge0/4
  description "VLAN tag = 27"
  no native-vlan
  no shutdown
 !
!
vEdge#
```

After your have configured an interface in a bridge domain, you add or change a VLAN identifier for that domain only by first deleting the bridge domain from the configuration (with a **no bridge** *bridge-id* command) and then reconfiguing the domain with the desired interface name and VLAN tag identifier.

To see which interfaces bridging is running on, use the **show bridge interface** command:

```
vEdge# show bridge interface

                         ADMIN   OPER    ENCAP                RX    RX      TX    TX
BRIDGE  INTERFACE  VLAN  STATUS  STATUS  TYPE   IFINDEX  MTU  PKTS  OCTETS  PKTS  OCTETS
-------------------------------------------------------------------------------------------
2       ge0/4      27    Up      Up      vlan   41       1500 4     364     0     0
```

"Up" in the Admin Status column indicates that the interface has been configured, and "Up" in the Oper Status column indicates that bridging is running on the interface.

Create a Bridge Domain with an Untagged VLAN

All frames in an untagged VLAN are sent with no VLAN tag, or VLAN ID, in the frame header. For frames that already contain a tag, the tag is removed before it is sent.

In the minimal configuration for a tagged VLAN, you simply create a bridging domain that contains an interface:

1. Create a bridging domain. This domain is identified by a unique integer.
   vEdge(config)# **bridge** *number*
   On each vEdge router, you can configure up to 16 bridging domains.

2. Associate an interface with the bridging domain, and enable that interface:
   vEdge(config-bridge)# **interface** *interface-name*
   vEdge(config-interface)# **no shutdown**

You can also configure the optional parameters described in the previous section.

Configure a Native VLAN

In the minimal configuration for a native VLAN, you create a bridging domain that contains an interface, and you mark that interface as a native VLAN interface:

1. Create a bridging domain. This domain is identified by a unique integer.
   vEdge(config)# **bridge** *number*
   On each vEdge router, you can configure up to 16 bridging domains.

2. Associate an interface with the bridging domain, and enable that interface:
   vEdge(config-bridge)# **interface** *interface-name*
   vEdge(config-interface)# **no shutdown**

3. Enabled native VLAN on the interface:
   vEdge(config-interface)# **native-vlan**

You can also configure the optional parameters described in the section about creating a tagged VLAN.

Configuring IRB

With bridging, all frame traffic remains within its VLAN. To allow frames to be passed among different VLANs, you enable integrated routing and bridging (IRB). To do this, you create a logical IRB interface in a VPN domain that connects to the bridge domain. Frames with destinations in other VLANs travel over the IRB interface to the VPN domain, and the Layer 3 route table is used to forward the frames toward their destination. The route table learns the routes to other IRB interfaces. With IRB, communication can be established between VLANs that are connected to the same VPN. The VLANs can be both on the local vEdge router and on a remote router.

In a minimal configuration to configure IRB, you create an IRB interface and assign it an IP address:

1. In the desired VPN, create an IRB interface:
   vEdge(config)# **vpn** *number*
   vEdge(config-vpn)# **interface irb** *number*
   The VPN number can be any number from 1 through 65530, which correspond to service VPNs, except for 512 (which is the management VPN). You cannot place IRB interfaces in either the transport VPN (VPN 0) or the management VPN (VPN 512).
   The IRB interface type is **irb** . The IRB interface number is a number from 1 through 63, and it must be the same number as the the identifier of the bridging domain that the IRB is connected to. For example, if you configure a bridging domain with an identifier of 2 (with the command **bridge 2**), the IRB interface number must be 2, and so you must configure **interface irb2** .

2. Configure an IP address for the IRB interface. This address is the subnet for the VLAN in the connected bridge domain:
   vEdge(config-irb)# **ip address** *prefix* **/** *length*

3. Enable the interface:
   vEdge(config-irb)# **no shutdown**

In all respects, the logical IRB interfaces is just another interface. This means, for instance, that you can configure additional interfaces properties as desired. (Note, however, that you cannot configuration autonegotiation on IRB interfaces.) It also means that you can ping a

logical IRB interface from another device in the same VPN, and you can ping the interface regardless of whether a corresponding bridge exists for that IRB interface. That is, if you configure interface **irb4** , but there is no corresponding **bridge 4** , you are still able to ping **irb4** .

Here is an example IRB configuration:

```
vEdge# show running-config vpn 1
vpn 1
 interface ge0/4
  ip address 10.20.24.15/24
  no shutdown
 !
 interface irb1
  ip address 1.1.1.15/24
  no shutdown
  access-list IRB_ICMP in
  access-list IRB_ICMP out
 !
 interface irb50
  ip address 3.3.3.15/24
  no shutdown
 !
!
vEdge# show running-config vpn 2
vpn 2
 interface irb2
  ip address 2.2.2.15/24
  no shutdown
 !
!
```

To display information about the IRB interfaces, use the **show interface** command. The IRB interfaces are listed in the Interface column, and the Encapsulation Type columns marks these interfaces as "vlan".

```
vEdge# show interface

                                    IF      IF
TCP
                                    ADMIN   OPER    ENCAP                                       SPEED
MSS                  RX      TX
VPN  INTERFACE  IP ADDRESS          STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR             MBPS
DUPLEX  ADJUST  UPTIME      PACKETS PACKETS
-------------------------------------------------------------------------------------------------------------
-------------------------------------
0    ge0/0      10.1.15.15/24       Up      Up      null   transport  1500  00:0c:29:cb:4f:9c  10     full
0       0:02:48:12  1467    1460
0    ge0/1      -                   Up      Up      null   service    1500  00:0c:29:cb:4f:a6  10     full
0       0:02:48:12  0       0
0    ge0/2      -                   Up      Up      null   service    1500  00:0c:29:cb:4f:b0  10     full
0       0:02:48:03  0       0
0    ge0/3      10.0.20.15/24       Up      Up      null   service    1500  00:0c:29:cb:4f:ba  10     full
0       0:02:48:12  0       0
0    ge0/5      -                   Up      Up      null   service    1500  00:0c:29:cb:4f:ce  10     full
0       0:02:48:03  0       0
0    ge0/6      -                   Up      Up      null   service    1500  00:0c:29:cb:4f:d8  10     full
0       0:02:48:03  0       0
0    ge0/7      10.0.100.15/24      Up      Up      null   service    1500  00:0c:29:cb:4f:e2  10     full
0       0:02:48:12  0       0
0    system     172.16.255.15/32    Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:02:48:12  0       0
1    ge0/4      10.20.24.15/24      Up      Up      null   service    1500  00:0c:29:cb:4f:c4  10     full
0       0:02:48:00  92      14
1    irb1       1.1.1.15/24         Up      Up      vlan   service    1500  00:0c:00:00:aa:00  10     full
0       0:02:48:00  1178    0
1    irb50      3.3.3.15/24         Up      Up      vlan   service    1500  00:0c:00:00:aa:00  10     full
0       0:02:48:00  0       0
```

```
2    irb2       2.2.2.15/24       Up      Up      vlan   service    1500  00:0c:00:00:aa:00  10     full
0       0:02:48:01  0       0
512  eth0       10.0.1.15/24      Up      Up      null   service    1500  00:50:56:00:01:05  1000   full
0       0:02:48:01  210     148
```

Additional Information

Bridging Overview

Bridging CLI Reference

CLI commands for configuring and monitoring Layer 2 bridging and Layer 3 integrated routing and bridging (IRB) on vEdge routers.

Bridging Configuration Commands

Use the following commands to configure bridging on a vEdge router.

```
bridge bridge-id
  age-time seconds
  interface interface-name
    description "text description"
    native-vlan
    [no] shutdown
    static-mac-address mac-address
  max-macs number
  name text
  vlan number
```

Bridging Monitoring Commands

Use the following commands to monitor Layer 2 bridging on a vEdge router:

- **clear bridge mac** — Clear the MAC addresses that the vEdge router has learned.
- **clear bridge statistics** —Clear the bridging statistics.
- **show bridge interface** —List information about the interfaces on which bridging is configured.
- **show bridge mac** —List the MAC addresses that the vEdge router has learned.
- **show bridge table** —List the information in the bridge forwarding table.

IRB Configuration Commands

Use the following commands to configure IRB within a VPN on a vEdge router:

```
vpn vpn-id
  interface irbnumber
    access-list acl-list
    arp
      ip address ip-address mac mac-address
    arp-timeout seconds
    autonegotiate
    clear-dont-fragment
    description "text description"
    dhcp-server (on vEdge routers only)
```

```
    address-pool prefix/length
    exclude ip-address
    lease-time minutes
    max-leases number
    offer-time minutes
    options
      default-gateway ip-address
      dns-servers ip-address
      domain-name domain-name
      interface-mtu mtu
      tftp-servers ip-address
    static-lease mac-address
  ip address address/subnet
  mac-address mac-address
  mtu bytes
  [no] shutdown
  tcp-mss-adjust bytes
```

IRB Monitoring Commands

Use the following commands to monitor IRB :

- **show interface** —List information about the interfaces on which IRB is enabled.

Additional Information

Configuring Bridging and IRB

Segmentation

Segmentation (VPN) Overview

This article illustrates the segmentation and VPN capabilities of the Viptela overlay network solution.

Network segmentation has existed for over a decade and has been implemented in multiple forms and shapes. At its most rudimentary level, segmentation provides traffic isolation. The most common forms of network segmentation are virtual LANs, or VLANs, for Layer 2 solutions, and virtual routing and forwarding, or VRF, for Layer 3 solutions.

There are many use cases for segmentation:

- An enterprise wants to keep different lines of business separate (for example, for security or audit reasons).
- The IT department wants to keep authenticated users separate from guest users.
- A retail store wants to separate video surveillance traffic from transactional traffic.
- An enterprise wants to give business partners selective access only to some portions of the network.
- A service or business needs to enforce regulatory compliance, such as compliance with HIPAA, the U.S. Health Insurance Portability and Accountability Act, or with the Payment Card Industry (PCI) security standards.
- A service provider wants to provide VPN services to its medium-sized enterprises.
- An enterprise wants to set up a trial run of new service and wants to use a cloud service for development and system test.

One inherent limitation of segmentation is its scope. Segmentation solutions either are complex or are limited to a single device or pair of devices connected via an interface. As an example, Layer 3 segmentation provides the following:

1. Ability to group prefixes into a unique route table (RIB or FIB).
2. Ability to associate an interface with a route table so that traffic traversing the interface is routed based on prefixes in that route table.

This is useful functionality, but the scope is limited to a single device. To extend the functionality throughout the network, the segmentation information needs to be carried to the relevant points in the network. There are two approaches to providing this network-wide segmentation:

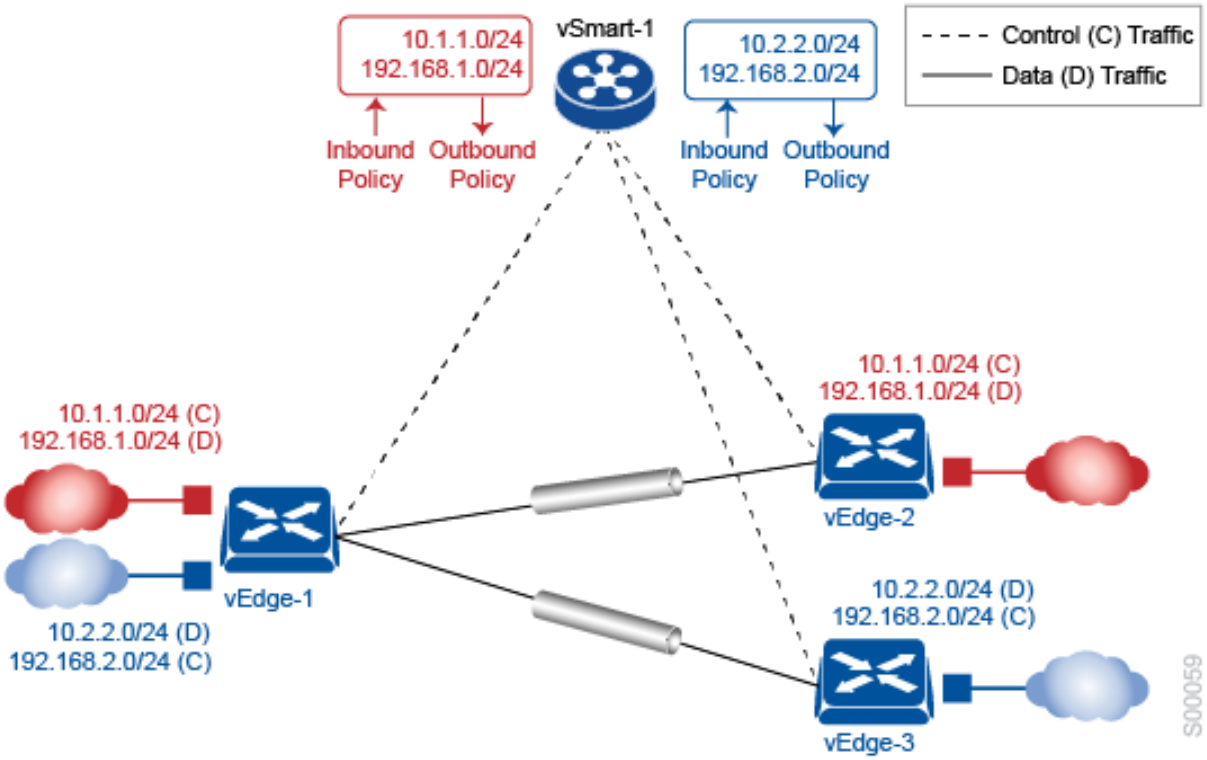
- Define the grouping policy at every device and on every link in the network (basically, you perform Steps 1 and 2 above on every device).
- Define the grouping policy at the edges of the segment, and then carry the segmentation information in the packets for intermediate nodes to handle.

The first approach is useful if every device is an entry or exit point for the segment, which is generally not the case in medium and large networks. The second approach is much more scalable and keeps the transport network free of segments and complexity. MPLS-based Layer 3 VPNs are a popular example of segmentation at the edge.

Segmentation using the Viptela Technology

Viptela employs the more prevalent and scalable model of creating segments. Essentially, segmentation is done at the edges, on a vEdge router, and the segmentation information is carried in the packets in the form of an identifier.

The figure below shows the propagation of routing information inside a VPN.



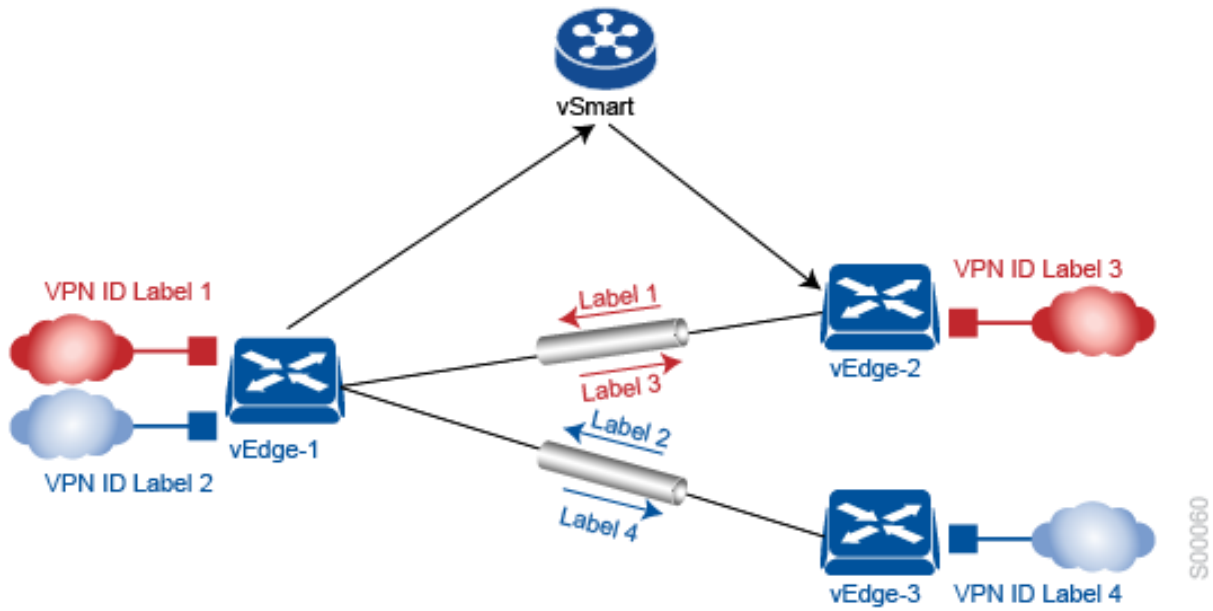
In this figure:

- vEdge-1 subscribes to two VPNs, red and blue.
  - The red VPN caters to the prefix 10.1.1.0/24 (either directly through a connected interface or learned via the IGP or BGP).
  - The blue VPN caters to the prefix 10.2.2.0/24 (either directly through a connected interface or learned via the IGP or BGP).
- vEdge-2 subscribes to the red VPN.
  - This VPN caters to the prefix 192.168.1.0/24 (either directly through a connected interface or learned via the IGP or BGP).
- vEdge-3 subscribes to the blue VPN.
  - This VPN caters to the prefix 192.168.2.0/24 (either directly through a connected interface or learned via the IGP or BGP).

Because each vEdge router has an OMP connection over a TLS tunnel to a vSmart controller, it propagates its routing information to the vSmart controller. On the vSmart controller, the network administrator can enforce policies to drop routes, to change TLOCs (which are overlay next hops) for traffic engineering or service chaining, or to change the VPN ID (see Policy Overview for more details). The network administrator can apply these policies as inbound and outbound policies on the vSmart controller.

All prefixes belonging to a single VPN are kept in a separate route table. This provides the Layer 3 isolation required for the various segments in the network. So, vEdge-1 has two VPN route tables, and vEdge-2 and vEdge-3 each have one route table. In addition, the vSmart controller maintains the VPN context of each prefix.

Separate route tables provide isolation on a single node. So now the question is how to propagate the routing information across the network. In the Viptela solution, this is done using VPN identifiers, as shown in the figure below. A VPN ID carried in the packet identifies each VPN on a link. When you configure a VPN on a vEdge router, the VPN has a label associated with it. The vEdge router sends the label, along with the VPN ID, to the vSmart controller. The vSmart controller propagates this vEdge-to-VPN-ID mapping information to the other vEdge routers in the domain. The remote vEdge routers then use this label to send traffic to the appropriate VPN. The local vEdge routers, on receiving the data with the VPN ID label, use the label to demultiplex the data traffic. This is similar to how MPLS labels are used. This design is based on standard RFCs and is compliant with regulatory procedures (such as PCI and HIPAA).



It is important to point out that the transport network that connects the vEdge routers is completely unaware of the VPNs. Only the vEdge routers know about VPNs; the rest of the network follows standard IP routing.

Default VPNs

The Viptela solution provides two default VPNs to separate traffic.

To enforce the inherent separation between services (such as prefixes that belong to the enterprise) and transport (the network that connects the vEdge routers), all the transport interfaces (that is, all the TLOCs) are kept in the transport VPN, which is internally maintained as VPN 0. This ensures that the transport network cannot reach the service network by default. Multiple transport interfaces can belong to the same transport VPN, and packets can be forwarded to and from transport interfaces.

Management ports are kept separate as well and maintain a separate VPN, which is internally maintained as VPN 512.

Dual Stack

In the transport VPN (VPN 0), vEdge routers and vSmart controllers support dual stack. To enable dual stack, configure an IPv4 address and an IPv6 address on the tunnel interface. The vEdge router learns from the vSmart controller whether a destination supports IPv4 or IPv6 addresses. When forwarding traffic, the router chooses either the IPv4 or the IPv6 TLOC based on the destination address.

Additional Information

Configuring Segmentation (VPNs)
Segmentation (VPN) Configuration Examples

Configuring Segmentation (VPNs)

In the Viptela overlay network, VPNs divide the network into different segments. By default, two VPNs are present in the configurations of all Viptela devices, and these VPNs serve specific purposes:

- VPN 0 is the transport VPN. It carries control traffic over secure DTLS or TLS connections between vSmart controllers and vEdge routers, and between vSmart controllers and vBond orchestrators. Initially, VPN 0 contains all a device's interfaces except for the management interface, and all the interfaces are disabled. For the control plane to establish itself so that the overlay network can function, you must configure WAN transport interfaces in VPN 0.
- VPN 512 is the management VPN. It carries out-of-band network management traffic among the Viptela devices in the overlay network. By default, VPN 512 is configured and enabled. You can modify this configuration if desired.

To segment user networks and user data traffic locally at each site and to interconnect user sites across the overlay network, you create additional VPNs on vEdge routers. (These VPNs are identified by a number that is not 0 or 512.) To enable the flow of data traffic, you associate interfaces with each VPN, assigning an IP address to each interface. These interfaces connect to local-site networks, not to WAN transport clouds. For each of these VPN, you can set other interface-specific properties, and you can configure features specific for the user segment, such as BGP and OSPF routing, VRRP, QoS, traffic shaping, and policing.

This article provides basic configuration procedures for the three types of VPNs. For more detailed information about configuring other VPN parameters, such as interface properties and routing protocols, refer to the Additional Information at the end of this article.

Configure the Transport VPN (VPN 0)

For the control plane to establish itself so that the overlay network can function, you must configure interfaces in VPN 0 to carry the control traffic necessary to establish and maintain the overlay network.

On vEdge routers, the interfaces in VPN 0 connect to some type of transport network or cloud, such as the Internet, MPLS, or Metro Ethernet. For each interface in VPN 0, you must set an IP address, and you create a tunnel connection that sets the color and encapsulation for the WAN transport connection. (The encapsulation is used for the transmission of data traffic.) These three parameters—IP address, color, and encapsulation—define a TLOC (transport location) on the vEdge router. The OMP session running on each tunnel sends the TLOC to the vSmart controllers so that they can learn the overlay network topology. For VPN 0, you can also set other interface-specific and VPN-specific properties in VPN 0.

Because vSmart controllers are responsible for determining the best routes through the overlay network (based on the TLOCs it learns and based on centralized policies), they handle only control plane traffic, in VPN 0. A vSmart controller can have only one interface in VPN 0, for which you set an IP address and you create a tunnel connection. This tunnel connection acts a control plane tunnel termination point.

Configure the Transport VPN on a vEdge Router

On a vEdge router, the interfaces in VPN 0 connect to a WAN transport network. You must configure at least one tunnel interface on a vEdge router so that it can join the control plane and be part of the overlay network. If is not configured, that router cannot participate in the overlay network.

For a tunnel connection on a vEdge router, you must configure the three components of a TLOC—the interface's IP address and the tunnel's color and encapsulation. An OMP session runs over each tunnel connection, and it is OMP that distributes the device TLOCs to vSmart controllers. The controllers use the TLOCs to determine the overlay network topology and to determine the best routing paths across the overlay network. A vEdge router can have up to four TLOCs, so you can configure more than one tunnel connection.

In the transport VPN (VPN 0), vEdge routers support dual stack. To enable dual stack, configure an IPv4 address and an IPv6 address on the tunnel interface. The vEdge router learns from the vSmart controller whether a destination supports IPv4 or IPv6 addresses. When forwarding traffic, the router chooses either the IPv4 or the IPv6 TLOC based on the destination address.

To configure VPN 0 on a vEdge router:

1. Configure the WAN transport interface:
   vEdge(config)# **vpn 0 interface** *interface-name*
   vEdge(config-interface)#
   In the most common cases, *interface-name* is the name of a physical Gigabit Ethernet interface (**ge** *port* **/** *slot*). The interface name can also be **gre** *number* , **ipsec** *number* , **loopback** *string* , **natpool** *number* , or **ppp** *number* .

2. Configure a static IPv4 address for the interface:
   vEdge(config-interface)# **ip address** *prefix / length*
   vEdge(config-interface)#
   Or you can enable DHCP on the interface so that the interface learn its IP address dynamically:
   vEdge(config-interface)# **ip dhcp-client** [**dhcp-distance** *number*]
   vEdge(config-interface)#
   When an interface learns its IPv4 address from a DHCP server, it can also learn routes from the server. By default, these routes have an administrative distance of 1, which is the same as static routes. To change the default value, include the **dhcp-distance** option, specifying a distance from 1 through 255.

3. To enable dual stack, configure a static IPv6 address for the interface:
   vEdge(config-interface)# **ipv6 address** *prefix / length*
   vEdge(config-interface)#
   Or you can enable DHCPv6 on the interface so that the interface learn its IP address dynamically:
   vEdge(config-interface)# **ipv6 dhcp-client** [**dhcp-distance** *number*] [**dhcp-rapid-commit**]
   vEdge(config-interface)#
   When an interface learns its IPv6 address from a DHCPv6 server, it can also learn routes from the server. By default, these routes have an administrative distance of 1, which is the same as static routes. To change the default value, include the **dhcp-distance** option, specifying a distance from 1 through 255. To speed up the assignment of IPv6 addresses, include the **dhcp-rapid-commit** option.

4. Enable the interface:
   vEdge(config-interface)# **no shutdown**

5. Configure the WAN transport tunnel connection:
   vEdge(config-interface)# **tunnel-interface**
   vEdge(config-tunnel-interface)#

6. Configure a color for the tunnel connection as an identifier for the tunnel:
   vEdge(config-tunnel-interface)# **color** *color*
   vEdge(config-tunnel-interface)#
   color can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . The default color is **default** .
   The colors **metro-ethernet** , **mpls** , and **private1** through **private6** are referred to as *private colors* , because they use private addresses to connect to the remote side vEdge router in a private network. You can use these colors in a public network provided that there is no NAT device between the local and remote vEdge routers.

7. Configure the encapsulation to use on tunnel connection:
   vEdge(config-tunnel-interface)# **encapsulation** (**gre** | **ipsec**)
   vEdge(config-tunnel-interface)#
   To configure both IPsec and GRE encapsulation, include two **encapsulation** commands. Note that if you do this, you are creating two TLOCs that have the same IP addresses and colors, but that have different encapsulation.

8. Configure any other properties specific to the tunnel interface, the interface, or VPN 0.

9. If you have a multi-TLOC environment, configure additional tunnel interfaces.

10. Enable DNS service in the VPN by configuring the IP address of a DNS server reachable from VPN 0:
    vEdge(config-vpn-0)# **dns** *ip-address* (**primary** | **secondary**)
    The address can be either an IPv4 or IPv6 address. By default, the IP address is for the primary DNS server.

11. If desired, configure IPv4 and IPv6 static routes in VPN 0:
    vEdge(config-vpn-0)# **ip route** *prefix* **/** *length next-hop* [*administrative-distance*]
    vEdge(config-vpn-0)# **ipv6 route** *prefix* **/** *length next-hop* [*administrative-distance*]

12. Activate the configuration:
    vEdge(config)# **commit**

To display interface information, use the **show interface** command for IPv4 interfaces and **show ipv6 interfaces** for IPv6 interfaces. To display information about DHCP and DHCPv6 servers, use the **show dhcp interface** and **show ipv6 dhcp interface** commands.

When you are troubleshooting routing and forwarding problems on a vEdge router, you can configure the router to perform route consistency checks, to determine whether the routes in the router's route and forwarding tables are consistent:

```
vEdge(config-system)# route-consistency-check
```

This command checks only IPv4 routes. Route consistency checking requires a large amount of device CPU, so it is recommended that you enable it only when you trouble shooting an issue and that you disable it at other times.

Here is an example of a VPN 0 configuration, where **interface ge0/0** is the WAN transport interface. This example shows that dual stack is enabled on the router, because the tunnel interface has both an IPv4 and an IPv6 address. Notice that the remaining seven device interfaces are part of VPN 0, because we have not yet configured any other VPNs. Also notice that the management interface is not present in VPN 0.

```
vpn 0
 interface ge0/0
  ip address 10.0.0.8/24
  ipv6 address fd00:1234::/16
  tunnel-interface
   color biz-internet
   encapslation ipsec
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
 interface ge0/1
  shutdown
 !
 interface ge0/2
  shutdown
 !
 interface ge0/3
  shutdown
 !
 interface ge0/4
  shutdown
 !
 interface ge0/5
  shutdown
 !
 interface ge0/6
  shutdown
 !
 interface ge0/7
  shutdown
 !
!
```

An interface can participate only in one VPN. So in an initial configuration, when VPN 0 is the only VPN that is configured, all the device's interfaces are present, by default, in VPN 0 (as shown in the output above). Then, when you create other VPNs to carry data traffic and configure interfaces in those VPNs, the interfaces used in the other VPNs are automatically removed from VPN 0. Here is an example in which **interface ge0/3** is used for VPN 1, so it has been automatically removed from the configuration of VPN 0:

```
vpn 0
interface ge0/0
  ip address 10.0.0.8/24
  tunnel-interface
   color biz-internet
   encapsulation ipsec
   allow-service dhcp
   allow-service dns
```

```
    allow-service icmp
    no allow-service sshd
    no allow-service ntp
    no allow-service stun
   !
   no shutdown
!
interface ge0/1
  shutdown
!
interface ge0/2
  shutdown
!
interface ge0/4
  shutdown
!
interface ge0/5
  shutdown
!
interface ge0/6
  shutdown
!
interface ge0/7
  shutdown
!
!
vpn 1
router
  ospf
   redistribute omp route-policy test-policy
   area 0
    interface ge0/3
    exit
   exit
  !
!
interface ge0/3
  ip address 10.10.10.1/24
  no shutdown
!
!
```

When you configure subinterfaces in a VPN that carries data traffic (that is, not VPN 0 and not VPN 512), the main interface must be configured with the **no shutdown** command so that it is enabled, and the main interface remains in VPN 0 once you configure the subinterface. For example, if in the VPN 1 configuration, you were to configure OSPF on VLAN 1, you can see that **interface ge0/3** remains present in VPN 0, while the subinterface **interface ge0/3.1** is used in VPN1:

```
vpn 0
 dns 1.2.3.4 primary
 interface ge0/0
  address 10.0.0.8/24
  tunnel-interface
   preference 100
   allow-service dhcp
   allow-service dns
   allow-service icmp
   allow-service sshd
   allow-service ntp
   allow-service stun
  !
  no shutdown
 !
 interface ge0/1
  shutdown
```

```
 !
 interface ge0/2
  shutdown
 !
 interface ge0/3
  no shutdown
 !
 interface ge0/4
  shutdown
 !
 interface ge0/5
  shutdown
 !
 interface ge0/6
  shutdown
 !
 interface ge0/7
  shutdown
 !
!
vpn 1
router
  ospf
   redistribute omp route-policy test-policy
   area 0
    interface ge0/3.1
    exit
   exit
  !
 !
interface ge0/3.1
  ip address 10.10.10.1/24
  no shutdown
 !
!
```

Configure the Transport VPN on a vSmart Controller

Because vSmart controllers are responsible for determining the best routes through the overlay network (based on the TLOCs it learns and based on centralized policies), they handle only control plane traffic, in VPN 0. A vSmart controller can have only one interface in VPN 0, for which you set an IP address and you create a tunnel connection. This tunnel connection acts a control plane tunnel termination point.

In the transport VPN (VPN 0), vEdge routers support dual stack. To enable dual stack, configure an IPv4 address and an IPv6 address on the tunnel interface. The vEdge router learns from the vSmart controller whether a destination supports IPv4 or IPv6 addresses. When forwarding traffic, the router chooses either the IPv4 or the IPv6 TLOC based on the destination address.

To configure VPN 0 on a vSmart controller:

1. Configure the WAN transport interface:
   vSmart(config)# **vpn 0 interface** *interface-name*
   vSmart(config-interface)#
   interface-name is the name of a virtual Ethernet interface (**eth** *number*).

2. Configure a static IPv4 address for the interface:
   vEdge(config-interface)# **ip address** *prefix* / *length*
   vEdge(config-interface)#
   Or you can enable DHCP on the interface so that the interface learn its IP address dynamically:
   vEdge(config-interface)# **ip dhcp-client** [**dhcp-distance** *number*]
   vEdge(config-interface)#
   When an interface learns its IPv4 address from a DHCP server, it can also learn routes from the server. By default, these routes have an

administrative distance of 1, which is the same as static routes. To change the default value, include the **dhcp-distance** option, specifying a distance from 1 through 255.

3. To enable dual stack, configure a static Pv6 address for the interface:
   vEdge(config-interface)# **ipv6 ddress** *prefix / length*
   vEdge(config-interface)#
   Or you can enable DHCPv6 on the interface so that the interface learn its IP address dynamically:
   vEdge(config-interface)# **ipv6 dhcp-client** [**dhcp-distance** *number*] [**dhcp-rapid-commit**]
   vEdge(config-interface)#
   When an interface learns its IPv6 address from a DHCPv6 server, it can also learn routes from the server. By default, these routes have an administrative distance of 1, which is the same as static routes. To change the default value, include the **dhcp-distance** option, specifying a distance from 1 through 255. To speed up the assignment of IPv6 addresses, include the **dhcp-rapid-commit** option.

4. Enable the interface:
   vSmart(config-interface)# **no shutdown**

5. Enable DNS service in the VPN by configuring the IP address of a DNS server reachable from VPN 0:
   vEdge(config-vpn-0)# **dns** *ip-address* (**primary** | **secondary**)
   The address can be either an IPv4 or IPv6 address. By default, the IP address is for the primary DNS server.

6. If desired, configure IPv4 and IPv6 static routes in VPN 0:
   vEdge(config-vpn-0)# **ip route** *prefix* **/** *length next-hop* [*administrative-distance*]
   vEdge(config-vpn-0)# **ipv6 route** *prefix* **/** *length next-hop* [*administrative-distance*]

7. Configure any other properties specific to the tunnel interface, the interface, or VPN 0.

8. Activate the configuration:
   vSmart(config)# **commit**

To display interface information, use the **show interface** command for IPv4 interfaces and **show ipv6 interfaces** for IPv6 interfaces. To display information about DHCP and DHCPv6 servers, use the **show dhcp interface** and **show ipv6 dhcp interface** commands.

Here is an example of a VPN 0 configuration on a vSmart controller:

```
vSmart# show running-config vpn 0
vpn 0
 dns 1.2.3.4 primary
 interface eth0
  ip dhcp-client
  no shutdown
 !
 interface eth1
  ip address 10.0.5.19/24
  tunnel-interface
   allow-ssh
   allow-icmp
  !
  no shutdown
 !
 ip route 0.0.0.0/0 10.0.5.13
!
```

Configure Data Traffic Exchange across Private WANs

When a vEdge router is connected to a private WAN, such as an MPLS or a metro Ethernet network, the carrier hosting the private network does not advertise the IP address of that vEdge router over the internet. (This IP address is associated with the TLOC on that vEdge router.) This means that remote vEdge routers are not able to learn how to reach that router and hence are not able to exchange data traffic with it directly over the private network.

To allow the vEdge router behind the private network to communicate directly over the private WAN with other vEdge routers, you direct the data traffic to a loopback interface rather than to the actual physical WAN interface. The overlay network can then advertise that the local router is reachable via its loopback address. To make it possible for the data traffic to actually be transmitted out the WAN interface, you bind the loopback interface to the physical WAN interface to the private network.

To configure VPN 0 so that it carries data traffic across private WANs:

1. Configure the loopback interface, assigning it an IP address:
   vEdge(config)# **vpn 0 loopback** *number* **ip address** *prefix* / *length*
   vEdge(config-loopback)# **no shutdown**

2. Configure the loopback interface to be a transport interface:
   vEdge(config-loopback)# **tunnel-interface**

3. Set the color of the loopback interface to be one of the primatel colors— **metro-ethernet** , **mpls** , and **private1** through **private6** . You must configure this same color on the loopback interfaces of all vEdge routers in the same private LAN.
   vEdge(config-tunnel-interface)# **color** *color*

Use the **show interface** command to check that the loopback interface in configured properly, as a transport interface with the proper IP address and color.

If a single vEdge router is connected to two (or more) different private networks, create a loopback interface for each private network, associate a carrier name with the interface so that the router can distinguish between the two private WANs, and "bind" the loopback interface to the physical interface that connects to the appropriate private WAN:

1. Configure the loopback interface, assigning it an IP address:
   vEdge(config)# **vpn 0 loopback** *number* **ip address** *prefix* / *length*
   vEdge(config-loopback)# **no shutdown**

2. Configure the loopback interface to be a transport interface and bind it to a physical interface:
   vEdge(config-loopback)# **tunnel-interface bind ge** *slot* / *port*

3. Configure a carrier name and TLOC color on the loopback interface:
   vEdge(config-tunnel-interface)# **carrier** *carrier-name*
   vEdge(config-tunnel-interface)# **color** *color*

4. On the physical interface, configure its IP address, and enable it:
   vEdge(config)# **vpn 0 interface ge** *slot* / *port* **ip address** *prefix* / *length*
   vEdge(config-ge)# **no shutdown**

Configure the Management VPN (VPN 512)

In the Viptela overlay network, VPN 512 is the network management VPN. It carries out-of-band management traffic in the overlay network. VPN 512 is configured and enabled by default on all Viptela devices. It contains the interface used for management traffic. For vEdge routers, this interface is generally a Gigabit Ethernet (**ge**) interface, and for other Viptela devices it is an **eth** interface. DHCP is enabled by default on the management interface. The default configuration for VPN 512 on a vEdge router looks like this:

```
vpn 512
 interface ge0/0
  ip dhcp-client
  no shutdown
 !
!
```

VPN 512 must be present on all Viptela devices so that they are always reachable on the network. You can configure additional parameters for VPN 512 if you choose.

Configure VPNs To Carry Data Traffic

VPNs other than VPN 0 and VPN 512 are used to carry data traffic across the overlay network. These VPNs are sometimes referred to as *service-side VPNs* . For these VPNs to operate, each one must have an operational interface (or subinterface). The remainder of what you configure in these VPNs depends on your network needs. You configure features specific for the user segment, such as BGP and OSPF routing, VRRP, QoS, traffic shaping, and policing.

To create a data traffic VPN:

1. Configure the VPN:
   vEdge(config)# **vpn** *number*
   vEdge(config-vpn)#
   The VPN number can be in the range 1 through 511, and 513 through 65535.

2. Configure at least one interface in the VPN and its IP address:
   vEdge(config-vpn)# **interface** *interface-name* **ip address** *address* / *prefix*
   vEdge(config-interface)#
   The interface name has the format **ge** *slot* / *port* , where the slot is generally 0 through 7 (depending on the device) and the port is 0 through 8. If you are configuring VLANs, specify a subinterface name in the format **ge** *slot* / *port* **.** *vlan* , where the VLAN number can be in the range 1 through 4094. (VLAN numbers 0 and 4095 are reserved.) The interface name can also be **gre** *number* , **ipsec** *number* , **loopback** *string* , **natpool** *number* , or **ppp** *number* .

3. Activate the interface:
   vEdge(config-interface)# **no shutdown**

4. Enable DNS service in the VPN by configuring the IP address of a DNS server reachable from that VPN:
   vEdge(config-vpn)# **dns** *ip-address*

5. If desired, configure IPv4 static routes in the VPN:
   vEdge(config-vpn)# **ip route** *prefix* **/** *length next-hop* [*administrative-distance*]

6. Configure any other properties specific to the interface or to VPN.

7. Activate the configuration:
   vEdge(config)# **commit**

Here is an example of a configuration for VPN 1:

```
vpn 1
 dns 1.2.3.4 primary
 router
  ospf
   redistribute omp route-policy test-policy
   area 0
    interface ge0/3
    exit
   exit
  !
 !
 interface ge0/3
  ip address 10.10.10.1/24
  no shutdown
 !
!
```

Dual-Stack Operation

When a Viptela device establishes an IPsec tunnel for control traffic between a local TLOC and a remote TLOC, or when a device establishes a BFD tunnel for data plane traffic between a local and a remote TLOC, an IPv6 tunnel is established in the following situations:

- The local device has only an IPv6 address, and the remote device has an IPv6 address.
- The remote device has only an IPv6 address, and the local device has an IPv6 address.

If both the local and remote devices have IPv4 addresses, IPsec and BFD always establish an IPv4 tunnel.

Additional Information

Configuring Interfaces
Configuring Multicast Overlay Routing
Configuring Unicast Overlay Routing
Segmentation (VPN) Overview
Segmentation (VPN) Configuration Examples

Segmentation (VPN) Configuration Examples

Some straightforward examples of creating and configuring VPNs to help you understand the configuration procedure for segmenting networks.

Create Basic VPNs

Creating the basic VPNs required by Viptela devices is a simple, straightforward process, consisting of these steps:

1. On the vEdge router:
   — Create a VPN instance for the transport VPN. VPN 0 is reserved for the transport VPN.
   — Create a VPN instance for the management VPN. VPN 512 is reserved for the management VPN.
   — Create a VPN instance to use for routing.

2. On the vSmart controller:
   — Create a VPN instance for the transport VPN. VPN 0 is reserved for the transport VPN.
   — Create a VPN instance for the management VPN. VPN 512 is reserved for the management VPN.
   — Optionally, create policies to influence routing and access control within the VPN.

Configuration on the vEdge Router

To create the basic VPNs on a vEdge router, you configure VPN 0 for transport, VPN 512 for management, and a third VPN (here, VPN 1) for carrying data traffic:

1. First, configure general system parameters:
   vEdge(config)# **system host-name** *host-name*
   vEdge(config-system)# **system-ip** *ip-address*
   vEdge(config-system)# **domain-id** *domain-id*
   vEdge(config-system)# **site-id** *site-id*
   vEdge(config-system)# **vbond** (*dns-name* | *ip-address*)

2. In VPN 0, which is the transport VPN, configure the interface to the WAN transport cloud, to establish reachability between the vEdge router and the vSmart controller, and between vEdge routers:

   a. Configure an IP address for the interface:
      vEdge(config)# **vpn 0 interface** *interface-name* **ip address** *prefix* / *length*

   b. Enable the interface:
      vEdge(config-interface)# **no shutdown**

   c. Enable a transport tunnel interface to carry control and data traffic, and configure the color and encapsulation for the tunnel:
      vEdge(config-interface)# **tunnel-interface**

vEdge(config-tunnel-interface)# **encapsulation** (**gre** | **ipsec**)
vEdge(config-tunnel-interface)# **color** *color*

d. Configure a default route for the VPN:
vEdge(config-vpn-0)# **ip route 0.0.0.0/0** *ip-address*

3. Configure a VPN for data traffic:

a. Create the VPN and assign it a identifier number. The identifier can be any number except 0 and 512.
vEdge(config)# **vpn** *vpn-id*

b. Add an interface to the VPN:
vEdge(config-vpn- *number*)# **interface** *interface-name* **ip address** *ip-address*

c. Enable the interface:
vEdge(config-vpn- *number*)# **no shutdown**

4. Configure unixAR routing in the VPN. See **Configuring Basic Unicast Overlay Routing** for more information.

5. Activate the configuration:
vEdge(config)# **commit**

Here is the full configuration on the vEdge router:

```
system                              # Configure general system parameters
 host-name vedge
 system-ip 1.0.0.2
 domain-id 1
 site-id   20
 vbond 10.2.6.1
!
vpn 0                               # Create the tunnel interface and allow
  interface ge 0/0                    reachability to vSmart in transport VPN
    ip address 10.2.6.11/24
    tunnel-interface
      color default
      encapsulation ipsec
   !
   no shutdown
   !
 ip route 0.0.0.0/0 10.2.6.12
!
vpn 1                               # Create new VPN, add interfaces and routing
 interface ge 0/1
  ip address 10.100.1.1/24
  no shutdown
 !
!
router
  bgp 20
   neighbor 10.100.1.2
    no shutdown
    remote-as 20
    address-family ipv4_unicast
    !
   !
  !
 !
vpn 512
  interface mgmt0
    ip dhcp-client
    no shutdown
```

```
  !
!
```

Configuration on the vSmart Controller

On the vSmart controller, you configure general system parameters and the two VPNs—VPN 0 for WAN transport and VPN 512 for network management—as you did for the vEdge router. Also, you generally create a centralized control policy that controls how the VPN traffic is propagated through the rest of the network. In this particular example, we create a central policy, shown below, to drop unwanted prefixes from propagating through the rest of the network. You can use a single vSmart policy to enforce policies throughout the network.

Here are the steps for creating the control policy on the vSmart controller:

1. Create a list of sites IDs for the sites where you want to drop unwanted prefixes:
   vSmart(config)# **policy lists site-list 20-30 site-id 20**
   vSmart(config-site-list-20-30)# **site-id 30**

2. Create a prefix list for the prefixes that you do not want to propagate:
   vSmart(config)# **policy lists prefix-list drop-list ip-prefix 10.200.1.0/24**

3. Create the control policy:
   vSmart(config)# **policy control-policy drop-unwanted-routes sequence 10 match route prefix-list drop-list**
   vSmart(config-match)# **top**
   vSmart(config)# **policy control-policy drop-unwanted-routes sequence 10 action reject**
   vSmart(config-action)# **top**
   vSmart(config)# **policy control-policy drop-unwanted-routes sequence 10 default-action accept**
   vSmart(config-default-action)# **top**

4. Apply the policy to prefixes inbound to the vSmart controller:
   vSmart(config)# **apply-policy site-list 20-30 control-policy drop-unwanted-routes in**

Here is the full policy configuration on the vSmart controller:

```
apply-policy
 site-list 20-30
  control-policy drop-unwanted-routes in
 !
!
policy
 lists
  site-list 20-30
   site-id 20
   site-id 30
  !
  prefix-list drop-list
   ip-prefix 10.200.1.0/24
  !
 !
 control-policy drop-unwanted-routes
  sequence 10
   match route
    prefix-list drop-list
   !
   action reject
   !
  !
  default-action accept
 !
!
```

Control VPN Membership

You can create VPNs just at the sites of interest and can then keep them hidden so that the rest of the network does not even know about them and the routes from them. Such a network design provides a great deal of traffic isolation and flexibility. However, there might be cases where the network administrator might want to explicitly disallow the creation of VPNs on the vEdge router. An example is in a B2B partnership, when the vEdge router is not located at the customer premise. For these situations, the network administrator can choose to allow only certain VPNs on these vEdge routers. Effectively, you are controlling membership in the VPN.

You control VPN membership policy at the vSmart controller. In the example here, you create a policy that explicitly disallows VPN 1 at sites 20 and 30:

```
apply-policy
 site-list 20-30
  vpn-membership disallow-vpn1
 !
!
policy
 lists
  site-list 20-30
   site-id 20
   site-id 30
  !
 !
 vpn-membership disallow-vpn1
  sequence 10
   match vpn-id 1
   action reject
   !
  !
  default-action accept
 !
!
```

Leak Routes across VPNs

In some situations it is desirable to leak routes from one VPN into another. Some examples include extranets, where you are making a portion of your intranet available to users outside your organization, B2B partnerships, and the network transition that occurs during a merger or acquisition. To leak routes across VPNs, you create a leaking control policy on the vSmart controller, a design that allows you to control route leaking from a central point in the network.

In this example, we create a control policy that allows an enterprise's VPN to import routes from a VPN list. Specifically, we:

- Create a control policy to match routes from a list of VPNs. Here, sequence 10 of the policy matches all routes from the VPNs of all business partners (BPs). The business partner VPN IDs are listed in the **All-BPs** list.
- Accept routes that match this policy, and import the prefixes into a new VPN called **Enterprise-BP** .
- Apply this policy towards the BP sites on vRoutes inbound to the vSmart controller.

```
policy
  lists
    site-list BP-Sites
      site-id 10
      site-id 20
    vpn-list All-BPs
      vpn 100
      vpn 101
    vpn-list Enterprise-BP
      vpn 200
  control-policy import-BPs-to-Enterprise
    sequence 10
```

```
        match route
         vpn-list All-BPs
        !
        action accept
         export-to vpn-list Enterprise-BP
         !
        !
       !
       default-action accept
      !
!
apply-policy
 site-list BP-Sites
  control-policy import-BPs-to-Enterprise in
 !
```

This policy matches all routes from all VPNs in the **All-BPs** VPN lists and populates these prefixes into the VPNs in the Enterprise-BP list. The routing table of the Enterprise-BP VPN will now contain all the prefixes of the BPs.

One advantage of importing routes in this way is access control. Keeping each BP in a separate VPN and creating an extranet policy ensures that the BPs cannot talk to each other.

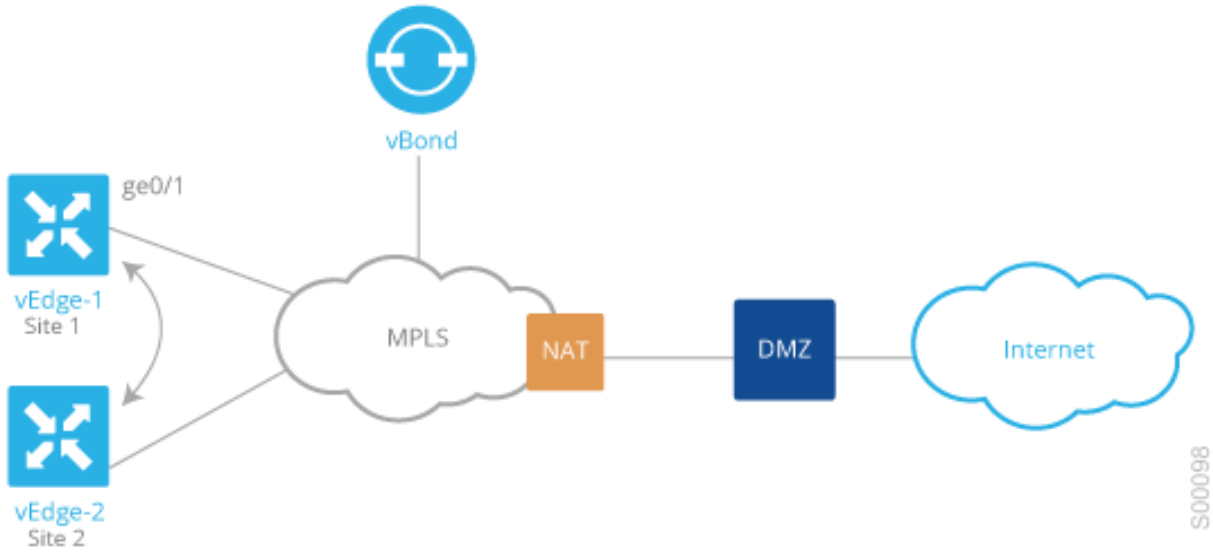
Allow Data Traffic Exchange across Private WANs

When the WAN network to which a vEdge router is connected is a private network, such as an MPLS or a metro Ethernet network, and when the carrier hosting the private network does not advertise the router's IP address, remote vEdge routers on the same private network but at different sites can never learn how to reach that router and hence are not able to exchange data traffic with it by going only through the private network. Instead, the remote routers must route data traffic through a local NAT and over the Internet to a vBond orchestrator, which then provides routing information to direct the traffic to its destination. This process can add significant overhead to data traffic exchange, because the vBond orchestrator may physically be located at a different site or a long distance from the two vEdge routers and because it may be situated behind a DMZ.

To allow vEdge routers at different overlay network sites on the private network to exchange data traffic directly using their private IP addresses, you configure their WAN interfaces to have one of the private colors, **metro-ethernet** , **mpls** , and **private1** through **private6** . Of these private colors, the WAN interfaces on the vEdge routers must be marked with the same color so that they can exchange data traffic.

Exchange Data Traffic within a Single Private WAN

To illustrate the exchange of data traffic across private WANs, let's look at a simple topology in which two vEdge routers are both connected to the same private WAN. The following figure shows that these two vEdge routers are connected to the same private MPLS network. The vEdge-1 router is located at Site 1, and vEdge-2 is at Site 2. Both routers are directly connected to PE routers in the carrier's MPLS cloud, and you want both routers to be able to communicate using their private IP addresses.



This topology requires a special configuration to allow traffic exchange using private IP addresses because:

- The vEdge routers are in different sites; that is, they are configured with different site IDs.
- The vEdge routers are directly connected to the PE routers in the carrier's MPLS cloud.
- The MPLS carrier does not advertise the link between the vEdge router and its PE router.

To be clear, if the situation were one of the following, no special configuration would be required:

- vEdge-1 and vEdge-2 are configured with the same site ID.
- vEdge-1 and vEdge-2 are in different sites, and the vEdge router connects to a CE router that, in turn, connects to the MPLS cloud.
- vEdge-1 and vEdge-2 are in different sites, the vEdge router connects to the PE router in the MPLS cloud, and the private network carrier advertises the link between the vEdge router and the PE router in the MPLS cloud.
- vEdge-1 and vEdge-2 are in different sites, and you want them to communicate using their public IP addresses.

In this topology, because the MPLS carrier does not advertise the link between the vEdge router and the PE router, you use a loopback interface on the each vEdge router to handle the data traffic instead of using the physical interface that connects to the WAN. Even though the loopback interface is a virtual interface, when you configure it on the vEdge router, it is treated like a physical interface: the loopback interface is a terminus for both a DTLS tunnel connection and an IPsec tunnel connection, and a TLOC is created for it.

This loopback interface acts as a transport interface, so you must configure it in VPN 0.

For the vEdge-1 and vEdge-2 routers to be able to communicate using their private IP addresses over the MPLS cloud, you set the color of their loopback interfaces to be the same and to one of private colors— **metro-ethernet** , **mpls** , and **private1** through **private6** .

Here is the configuration on vEdge-1:

```
vedge-1(config)# vpn 0
vedge-1(config-vpn-0)# interface loopback1
vedge-1(config-interface-loopback1)# ip address 172.16.255.25/32
vedge-1(config-interface-loopback1)# tunnel-interface
vedge-1(config-tunnel-interface)# color mpls
vedge-1(config-interface-tunnel-interface)# exit
vedge-1(config-tunnel-interface)# no shutdown
vedge-1(config-tunnel-interface)# commit and-quit
vedge-1# show running-config vpn 0
```

```
...
 interface loopback1
  ip-address 172.16.255.25/32
  tunnel-interface
   color mpls
  !
  no shutdown
 !
```

On vEdge-2, you configure a loopback interface with the same tunnel interface color that you used for vEdge-1:

```
vedge-2# show running-config vpn 0
vpn 0
 interface loopback2
  ip address 172. 17.255.26/32
  tunnel-interface
   color mpls
  no shutdown
 !
```

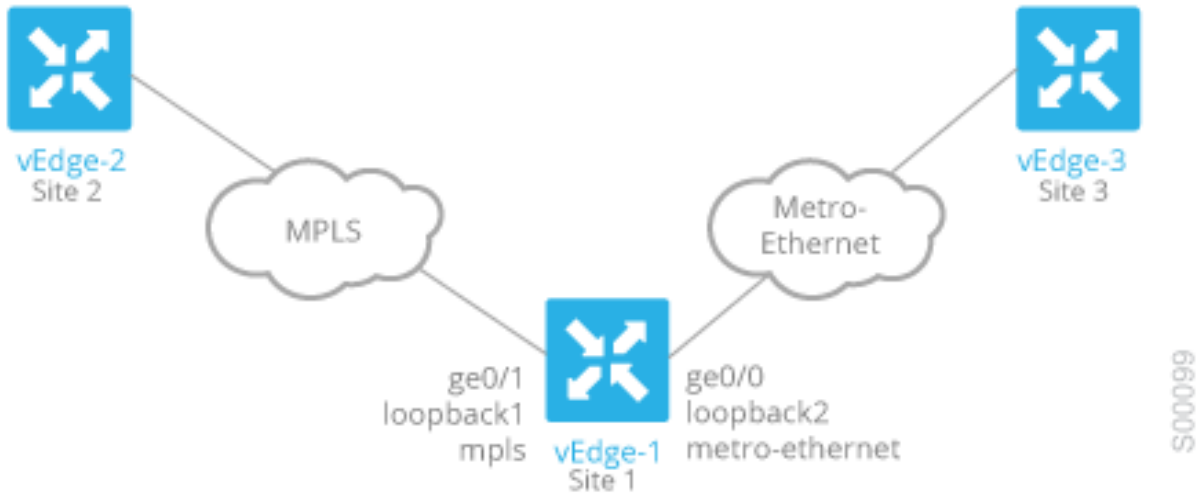
Use the **show interface** command to verify that the loopback interface is up and running. The output shows that the loopback interface is operating as a transport interface, so this is how you know that it is sending and receiving data traffic over the private network.

```
vedge-1# show interface
                                    IF      IF
TCP
                                    ADMIN   OPER    ENCAP                                       SPEED
MSS                  RX       TX
VPN  INTERFACE  IP ADDRESS          STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR             MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------
0    ge0/0      10.1.15.15/24       Up      Up      null   transport  1500  00:0c:29:7d:1e:fe  10     full
0       0:07:38:49  213199   243908
0    ge0/1      10.1.17.15/24       Up      Up      null   service    1500  00:0c:29:7d:1e:08  10     full
0       0:07:38:49  197      3
0    ge0/2      -                   Down    Down    null   service    1500  00:0c:29:7d:1e:12  -      -
0       -           1        1
0    ge0/3      10.0.20.15/24       Up      Up      null   service    1500  00:0c:29:7d:1e:1c  10     full
0       0:07:38:49  221      27
0    ge0/6      57.0.1.15/24        Up      Up      null   service    1500  00:0c:29:7d:1e:3a  10     full
0       0:07:38:49  196      3
0    ge0/7      10.0.100.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:44  10     full
0       0:07:44:47  783      497
0    loopback1  172.16.255.25/32    Up      Up      null   transport  1500  00:00:00:00:00:00  10     full
0       0:00:00:20  0        0
0    system     172.16.255.15/32    Up      Up      null   loopback   1500  00:00:00:00:00:00  10     full
0       0:07:38:25  0        0
1    ge0/4      10.20.24.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:26  10     full
0       0:07:38:46  27594    27405
1    ge0/5      56.0.1.15/24        Up      Up      null   service    1500  00:0c:29:7d:1e:30  10     full
0       0:07:38:46  196      2
512  eth0       10.0.1.15/24        Up      Up      null   service    1500  00:50:56:00:01:05  1000   full
0       0:07:45:55  15053    10333
```

To allow vEdge routers at different overlay network sites on the private network to exchange data traffic directly, you use a loopback interface on the each vEdge router to handle the data traffic instead of using the physical interface that connects to the WAN. You associate the same tag, called a carrier tag, with each loopback interface so that all the routers learn that they are on the same private WAN. Because the loopback interfaces are advertised across the overlay network, the vEdge routers are able to learn reachability information, and they can exchange data traffic over the private network. To allow the data traffic to actually be transmitted out the WAN interface, you bind the loopback interface to a physical WAN interface, specifically to the interface that connects to the private network. Remember that this is the interface that the private network does not advertise. However, it is still capable of transmitting data traffic.

Exchange Data Traffic between Two Private WANs

A variant of the topology illustrated above is the case in which a single vEdge router connects to two different private WANs, such as two different MPLS clouds provided by two different network carriers, or two different types of private WANs, as illustrated below. In this figure, the vEdge-1 router connects to one MPLS private WAN and one metro-Ethernet private WAN.



As in the previous example, you create loopback interfaces on the three routers. For vEdge-1, which connects to both of the private WANs, you create two loopback interfaces. For each one, you assign a color, as in the previous example. But you configure two more things: you assign a tag to identify the carrier, and you "bind" the loopback interface to the physical interface that connects to the private WAN. So, vEdge-1 has two loopback interfaces with these properties:

- Loopback1 has the color **mpls** , the carrier **carrier2** , and binds to physical interface ge0/1.
- Loopback 2 has the color **metro-ethernet** and the carrier **carrier1** , and binds to physical interface ge0/0.

The vEdge-2 router has a single loopback interface that connects to the MPLS private WAN. Its color is **mpls** , and its carrier is **carrier2** . Both these properties match those on the loopback1 interface on vEdge-1. However, because vEdge-2 connects to only one private WAN, there is no need to bind its loopback interface to a physical interface.

Finally, vEdge-3 has a single loopback interface with color **metro-ethernet** and carrier **carrier1** , matching the properties configured on the vEdge-1 loopback2 interface.

On vEdge-1, the configuration in VPN 0 looks like this:

```
vpn 0
 interface ge0/0
  ip address 10.1.15.15/24
  no shutdown
 !
 interface loopback2
  ip address 172.16.15.15/24
  tunnel-interface
   color   metro-ethernet
   carrier carrier1
   bind    ge0/0
  !
  no shutdown
 !
```

```
 interface ge0/1
  ip address 10.1.17.15/24
  no shutdown
 !
 interface loopback1
  ip address 172.16.17.15/24
  tunnel-interface
   color   mpls
   carrier carrier2
   bind    ge0/1
  !
  no shutdown
!
```

If you need to apply control policy to a particular private network, use the **match carrier** option when creating the control policy.

Share a Common Service across Different VPNs

When services such as firewalls or load balances are spread across multiple VPNs, you can create a policy that forces traffic from one VPN to use the services in another VPN. See the service control examples in Service Chaining Configuration Examples .

Additional Information

Segmentation (VPN) Overview
Configuring Segmentation (VPNs)

Segmentation CLI Reference

CLI commands for configuring and monitoring segmentation (VPNs).

Segmentation Configuration Commands

Use the following commands to configure segmentation on a vEdge router.

```
vpn vpn-id
  bandwidth-downstream kbps (on vEdge routers and vManage NMSs only)
  bandwidth-upstream kbps (on vEdge routers and vManage NMSs only)
  dns ip-address [primary | secondary]
  ecmp-hash-key layer4 (on vEdge routers only)
  host hostname ip ip-address
  interface interface-name
    access-list acl-list (on vEdge routers only)
    arp
      ip ip-address mac mac-address
    arp-timeout seconds (on vEdge routers only)
    autonegotiate (on vEdge routers only)
    block-non-source-ip (on vEdge routers only)
    clear-dont-fragment
    dead-peer-detection interval seconds retries number (on vEdge routers only)
    description text
    dhcp-helper ip-address (on vEdge routers only)
    dhcp-server (on vEdge routers only)
      address-pool prefix/length
      exclude ip-address
      lease-time seconds
      max-leases number
      offer-time minutes
      options
        default-gateway ip-address
```

```
    dns-servers ip-address
    domain-name domain-name
    interface-mtu mtu
    tftp-servers ip-address
  static-lease mac-address ip ip-address host-name hostname
dot1x
  accounting-interval seconds
  acct-req-attr attribute-number (integer integer | octet octet | string string)
  auth-fail-vlan vlan-id
  auth-order (mab | radius)
  auth-reject-vlan vlan-id
  auth-req-attr attribute-number (integer integer | octet octet | string string)
  control-direction direction
  das
    client ip-address
    port port-number
    require-timestamp
    secret-key password
    time-window seconds
    vpn vpn-id
  default-vlan vlan-id
  guest-vlan vlan-id
  host-mode (multi-auth | multi-host | single-host)
  mac-authentication-bypass
    allow mac-addresses
    server
  nas-identifier string
  nas-ip-address ip-address
  radius-servers tag
  reauthentication minutes
  timeout
    inactivity minutes
  wake-on-lan
dead-peer-detection interval time-units retries number (on vEdge routers only)
duplex (full | half)
flow-control (bidirectional | egress | ingress)
ike (on vEdge routers only)
  authentication-type type
    local-id id
    pre-shared-secret password
    remote-id id
  cipher-suite suite
  group number
  mode mode
  rekey seconds
  version number
(ip address ipv4-refix/length | ip dhcp-client [dhcp-distance number])
(ipv6 address ipv6-refix/length | ipv6 dhcp-client [dhcp-distance number] [dhcp-rapid-commit])
ip address-list prefix/length (on vSmart controller containers only)
ipsec (on vEdge routers only)
  cipher-suite suite
  perfect-forward-secrecy pfs-setting
  rekey seconds
  replay-window number
keepalive seconds retries (on vEdge routers only)
mac-address mac-address
mtu bytes
nat (on vEdge routers only)
  block-icmp-error
  direction (inside | outside)
  [no] overload
  port-forward port-start port-number1 port-end port-number2
    proto (tcp | udp) private-ip-address ip address private-vpn vpn-id
  refresh (bi-directional | outbound)
```

```
        respond-to-ping
        static source-ip ip-address1 translate-ip ip-address2 (inside | outside)
        tcp-timeout minutes
        udp-timeout minutes
      pmtu (on vEdge routers only)
      policer policer-name (on vEdge routers only)
      ppp (on vEdge routers only)
        ac-name name
        authentication (chap | pap) hostname name password password
      pppoe-client (on vEdge routers only)
        ppp-interface name
      profile profile-id (on vEdge routers only)
      qos-map name (on vEdge routers only)
      rewrite-rule name (on vEdge routers only)
      secondary-address ipv4-address (on vEdge routers only)
      shaping-rate name (on vEdge routers only)
      [no] shutdown
      speed speed
      static-ingress-qos number (on vEdge routers only)
      tcp-mss-adjust bytes
      technology technology (on vEdge routers only)
      tloc-extension interface-name (on vEdge routers only)
      tunnel-interface
        allow-service service-name
        bind geslot/port (on vEdge routers only)
        carrier carrier-name
        color color [restrict]
        connections-limit number
        encapsulation (gre | ipsec) (on vEdge routers only)
          preference number
          weight number
        hello-interval milliseconds
        hello-tolerance seconds
        last-resort-circuit (on vEdge routers only)
        low-bandwidth-link (on vEdge routers only)
        max-control-connections number (on vEdge routers only)
        nat-refresh-interval seconds
        port-hop
        vbond-as-stun-server (on vEdge routers only)
        vmanage-connection-preference number (on vEdge routers only)
      tunnel-destination ip-address (GRE interfaces; on vEdge routers only)
      tunnel-destination (dns-name | ipv4-address) (IPsec interfaces; on vEdge routers only)
      (tunnel-source ip-address | tunnel-source-interface interface-name) (GRE interfaces; on vEdge routers
only)
      (tunnel-source ip-address | tunnel-source-interface interface-name) (IPsec interfaces; on vEdge
routers only)
      upgrade-confirm minutes
      vrrp group-name (on vEdge routers only)
        priority number
        timer seconds
        track-omp
    ! end vpn interface
    ip route ip-address/subnet next-hop-address
    name text
    omp
      advertise (aggregate prefix [aggregate-only] | bgp | connected | network prefix | ospf type | static)
(on vEdge routers only)
    router (on vEdge routers only)
      bgp...
      igmp...
      multicast-replicator local
        threshold number
      ospf...
      pim...
    service service-name address ip-address (on vEdge routers only)
```

Segmentation Monitoring Commands

Use the following commands to monitor segmentation:

show bgp commands
show interface commands
show ospf commands

Additional Information

Segmentation (VPN) Overview
Segmentation (VPN) Configuration Examples
Service Chaining

Security

Security Overview

Security is a critical element of today's networking infrastructure. Network administrators and security officers are hard pressed to defend their network against attacks and breaches. As a result of hybrid clouds and remote employee connectivity, the security perimeter around networks is disappearing. There are multiple problems with the traditional ways of securing networks, including:

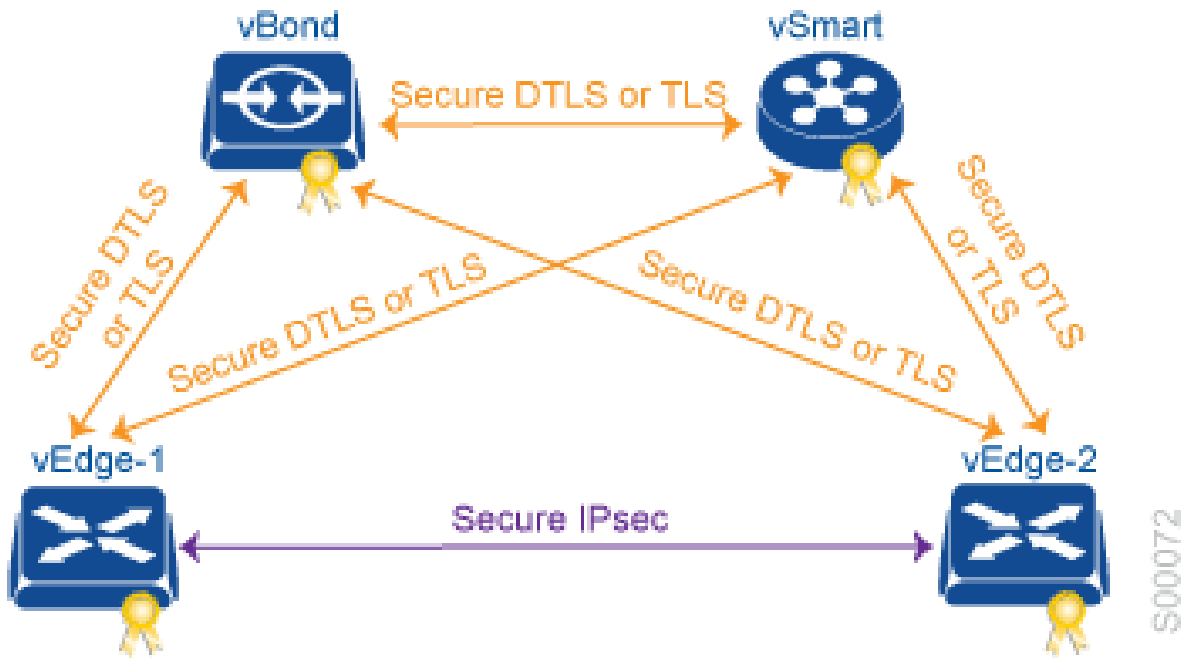
- Very little emphasis is placed on ensuring the authenticity of the devices involved in the communication.
- Securing the links between a pair of devices involves tedious and manual setup of keys and shared passwords.
- Scalability and high availability solutions are often at odds with each other.

Viptela Security Components

The Viptela solution takes a fundamentally different approach to security, basing its core design around the following precepts:

- Authentication—The solution ensures that only authentic devices are allowed to send traffic to one another.
- Encryption—All communication between each pair of devices is automatically secure, completely eliminating the overhead involved in securing the links.
- Integrity—No group keys or key server issues are involved in securing the infrastructure.

These three components—authentication, encryption, and integrity—are key to securing the Viptela overlay network infrastructure.



The articles on Control Plane Security Overview and Data Plane Security Overview examine how authentication, encryption, and integrity are implemented throughout the Viptela overlay network. The security discussion refers to the following illustration of the components of the Viptela network—the vSmart controller, the vBond orchestrator, and the vEdge routers. The connections between these devices form the control plane (in orange) and the data plane (in purple), and it is these connections that need to be protected by appropriate measures to ensure the security of the network devices and all network traffic.

Security Provided by NAT Devices

While the primary purpose of NAT devices is to allow devices with private IP addresses in a local-area network (LAN) to communicate with devices in public address spaces, such as the Internet, NAT devices also inherently provide a level of security, functioning as hardware firewalls to prevent unwanted data traffic from passing through the Viptela edge routers and to the LAN networks in the service-side networks connected to the vEdge router.

To enhance the security at branch sites, you can place the vEdge router behind a NAT device. The vEdge router can interact with NAT devices configured with the following Session Traversal Utilities for NAT (STUN) methods, as defined in RFC 5389 :

- Full-cone NAT, or one-to-one NAT—This method maps an internal address and port pair to an external address and port. Any external host can send packets to LAN devices behind the vEdge router by addressing them to the external address and port.
- Address-restricted cone NAT, or restricted-cone NAT—This method also maps an internal address and port to and external address and port. However, an external host can send packets to the internal device only if the external address (and any port at that address) has received a packet from the internal address and port.
- Port-restricted cone NAT—This method is a stricter version of restricted-cone NAT, in which an external host can send packets to the internal address and port only if the external address and port pair has received a packet from that internal address and port. The external device must send packets from the specific port to the specific internal port.
- Symmetric NAT—With this method, each request from the same internal IP address and port to an external IP address and port is mapped to a unique external source IP address and port. If the same internal host sends a packet with the same source address and port but to a different destination, the NAT device creates a different mapping. Only an external host that receives a packet from an internal host can send a packet back. vEdge routers support symmetric NAT only on one side of the WAN tunnel. That is, only one of the NAT devices at either end of the tunnel can use symmetric NAT.
  When a vEdge router operates behind a NAT device running symmetric NAT, only one of the NAT devices at either end of the tunnel can use symmetric NAT. The vEdge router that is behind a symmetric NAT cannot establish a BFD tunnel with a remote vEdge router that is behind a symmetric NAT, an address-restricted NAT, or a port-restricted NAT.
  To allow a vEdge router to function behind a symmetric NAT, you must configure the vManage NMS and vSmart controller control connections to use TLS. DTLS control connections do not work through a symmetric NAT.

Security for Connections to External Devices

Viptela vEdge routers can use the standards-based Internet Key Exchange (IKE) protocol when establishing IPsec tunnels between a device within the overlay network and a device that is external to the overlay network, such as a cloud-hosted service or a remote user. The Viptela software supports IKE version 2, which performs mutual authentication and establishes and maintains security associations (SAs). IPsec provides confidentiality, data integrity, access control, and data source authentication for the traffic being exchanged over the IPsec tunnel.

Additional Information

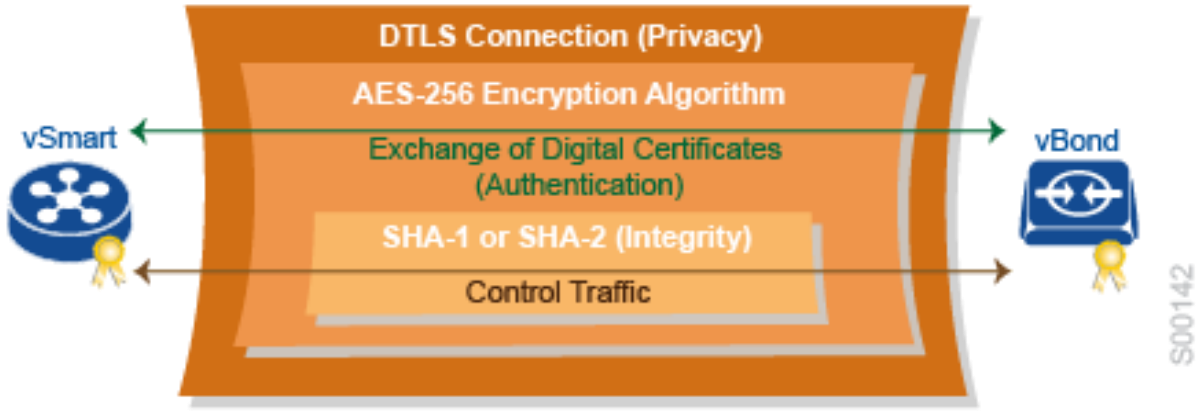
Configuring Security Parameters
Configuring IKE
Control Plane Security Overview
Data Plane Security Overview

Control Plane Security Overview

The control plane of any network is concerned with determining the network topology and defining how to direct packets. In a traditional network, the control plane operations of building and maintaining routing and forwarding tables and directing packets towards their destination are handled by routing and switching protocols, which typically offer few or no mechanisms for authenticating devices or for encrypting routing updates and other control information. In addition, the traditional methods for providing security are highly manual and do not scale. As examples, certificates are typically installed manually rather than in an automated fashion, and using preshared keys is not a very secure approach for providing device security.

The Viptela control plane has been designed with network and device security in mind. The foundation of the control plane is one of two security protocols derived from SSL (Secure Sockets Layer)—the Datagram Transport Layer Security (DTLS) protocol and the Transport Layer Security (TLS) protocol. The vSmart controller, which is the centralized brain of the Viptela solution, establishes and maintains DTLS or TLS connections to all Viptela devices in the overlay network: to the vEdge routers, the vBond orchestrators, to vManage NMSs, and to other vSmart controllers. These connections carry control plane traffic. DTLS or TLS provides communication privacy between Viptela devices in the network, using the Advanced Encryption Standard (AES-256) encryption algorithm to encrypt all control traffic sent over the connections.

The privacy and encryption in the control plane offered by DTLS and TLS provide a safe and secure foundation for the other two security components, authentication and integrity. To perform authentication, the Viptela devices exchange digital certificates. These certificates, which are either installed by the software or hard-coded into the hardware, depending on the device, identify the device and allow the devices themselves to automatically determine which ones belong in the network and which are imposters. For integrity, the DTLS or TLS connections run SHA-1 or SHA-2, a cryptographic secure hash algorithm which ensures that all control and data traffic sent over the connections has not been tampered with.



The following are the control plane security components, which function in the privacy provided by DTLS or TLS connections:

- **AES-256** encryption algorithm provides encryption services.
- **Digital certificates** are used for authentication.
- **SHA-1 or SHA-2** is responsible for ensuring integrity.

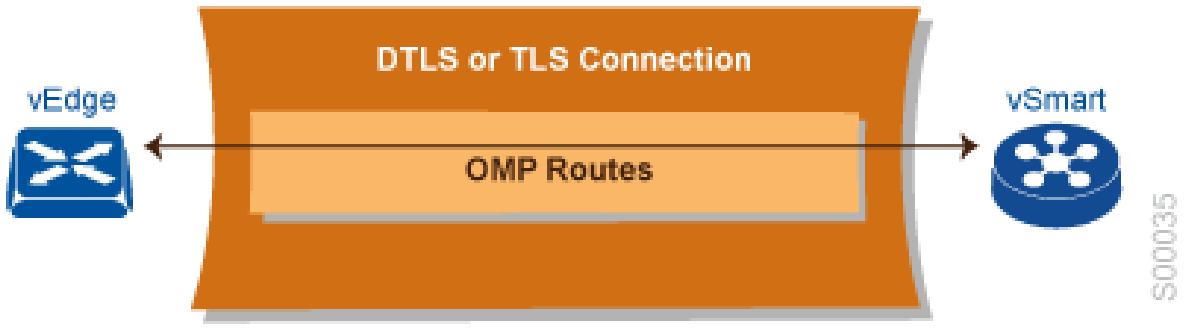
DTLS and TLS Infrastructure

Security protocols derived from SSL provide the foundation for the Viptela control plane infrastructure.

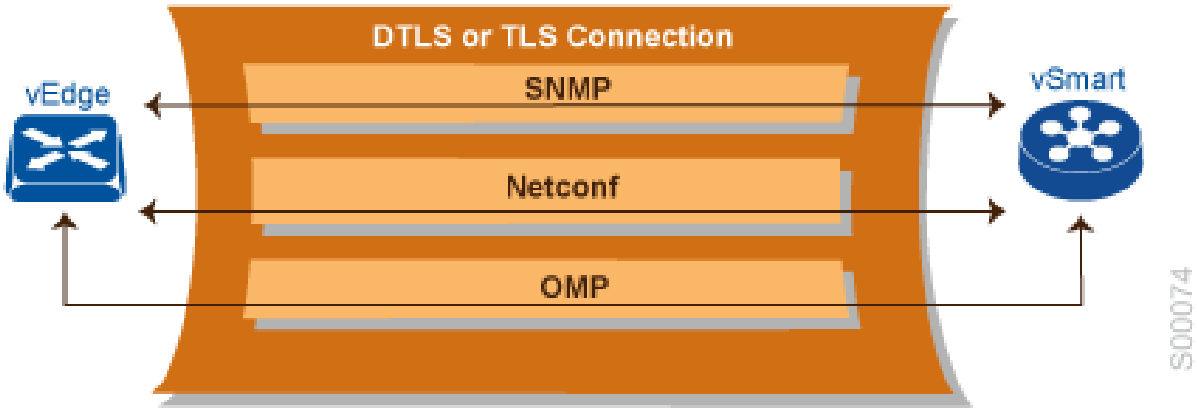
The first is the DTLS protocol, which is a transport privacy protocol for connectionless datagram protocols such as UDP, provides the foundation for the Viptela control plane infrastructure. It is based on the stream-oriented Transport Layer Security (TLS) protocol, which provides security for TCP-based traffic. (TLS itself evolved from SSL.) The Viptela infrastructure design uses DTLS running over UDP to avoid some of the issues with TCP, including the delays associated with stream protocols and some security issues. However, because UDP performs no handshaking and sends no acknowledgments, DTLS has to handle possible packet re-ordering, loss of datagrams, and data larger than the datagram packet size.

The control plane infrastructure can also be configured to run over TLS. This might be desirable in situations where the protections of TCP outweigh its issues. For example, firewalls generally offer better protection for TCP servers than for UDP servers.

The Viptela software implements the standard version of DTLS with UDP, which is defined in RFC 6347 . DTLS for use with other protocols is defined in a number of other RFCs . For TLS, the Viptela software implements the standard version defined in RFC 5246 .



In the Viptela architecture, the Viptela devices use DTLS or TLS as a tunneling protocol, which is an application-level (Layer 4) tunneling protocol. When the vSmart controllers, vBond orchestrators, vManage NMSs, and vEdge routers join the network, they create provisional DTLS or TLS tunnels between them as part of the device authentication process. After the authentication process completes successfully, the provisional tunnels between the vEdge routers and vSmart controllers, and those between the vBond orchestrators and vSmart controllers, become permanent and remain up as long as the devices are active in the network. It is these authenticated, secure DTLS or TLS tunnels that are used by all the protocol applications running on the Viptela devices to transport their traffic. For example, an OMP session on a vEdge router communicates with an OMP session on a vSmart controller by sending plain IP traffic through the secure DTLS or TLS tunnel between the two devices. (The Overlay Management Protocol is the Viptela control protocol used to exchange routing, policy, and management information among Viptela devices, as described in **Overlay Routing Overview** .)



A Viptela daemon running on each vSmart controller and vEdge router creates and maintains the secure DTLS or TLS connections between the devices. This daemon is called vdaemon and is discussed later in this article. After the control plane DTLS or TLS connections are established between these devices, multiple protocols can create sessions to run and route their traffic over these connections—including OMP, Simple Network Management Protocol (SNMP), and Network Configuration Protocol (Netconf)—without needing to be concerned with any security-related issues. The session-related traffic is simply directed over the secure connection between the vEdge routers and vSmart controllers.

Control Plane Authentication

The Viptela control plane uses digital certificates with 2048-bit RSA keys to authenticate the Viptela devices in the network. The digital certificates are created, managed, and exchanged by standard components of the public key infrastructure, or PKI:

- **Public keys** —These keys are generally known.
- **Private keys** —These keys are private. They reside on each Viptela device and cannot be retrieved from the device.

- **Certificates** signed by a root certification authority (CA)—The trust chain associated with the root CA needs to be present on all Viptela devices.

In addition to standard PKI components, the Viptela device serial numbers and the vEdge router chassis numbers are used in the authentication processes.

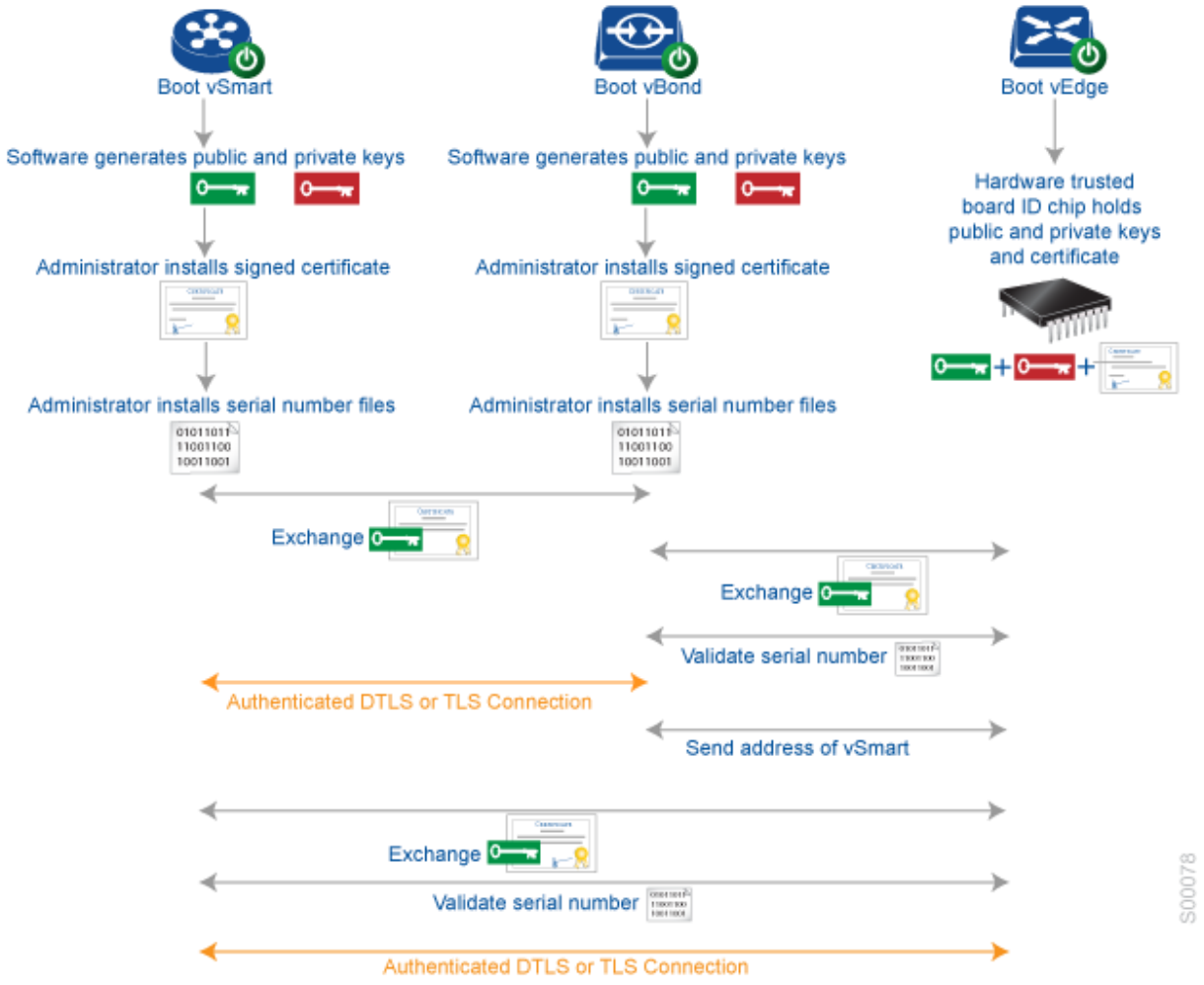
Let's first look at the PKI components that are involved in device authentication. On vEdge routers, the public and private keys and the certificates are managed automatically, by a Trusted Board ID chip that is built into the router. When the routers are manufactured, this chip is programmed with a signed certificate, which includes the device's public key and its serial number, and the device's private key. When the vEdge routers boot up and join the network, they exchange their certificates (including the device's public key and serial number) with other Viptela devices as part of the device authentication process. For networks with thousands or tens of thousands of vEdge routers, providing an automated process for managing keys and certificates greatly simplifies the task of maintaining security across the edge devices in the network. (Note that the vEdge router's private key always remains embedded in the router's Trusted Board ID chip, and it is never distributed, nor can it ever be retrieved from the device. In fact, any brute-force attempt to read the private key causes the Trusted Board ID chip to fail, thereby disabling all access to the router.)

For vSmart controllers, vBond orchestrators, and vManage NMS systems, the public and private keys and the certificates are managed manually. When you boot these devices for the first time, the Viptela software generates a unique private key–public key pair for each software image. The public key needs to be signed by the CA root. The network administrator then requests a signed certificate and manually installs it and the certificate chains on the vSmart controllers, vBond orchestrators, and vManage NMS systems. A typical network might have only a small handful of vSmart controllers, vBond orchestrators, and vManage NMSs, so the burden of manually managing the keys and certificates on these devices is small.

To augment these standard PKI components, the Viptela software uses the device serial numbers in performing automatic device authentication. Specifically, it uses the vEdge and vSmart serial numbers and the vEdge chassis numbers. When a batch of vEdge routers is shipped, the manufacturer sends a text file that lists the serial numbers of the vEdge routers and the corresponding chassis numbers. For the vSmart controllers, when the network administrator receives the signed certificate, they should extract the serial numbers from the certificates and place them into a single text file, one serial number per line. Then the network administrator manually installs these two files. The file received from the manufacturer that lists all valid vEdge serial and chassis numbers is uploaded and installed on vSmart controllers. Both the vEdge authorized serial number file and the file listing the vSmart serial numbers are uploaded and installed on vBond orchestrators. Then, during the automatic authentication process, as pairs of devices are establishing DTLS control connections between them, each device compares the serial numbers (and for vEdge routers, the chassis numbers) to those in the files installed on the device. A devices allows a connection to be established only if the serial number or serial–chassis number combination (for a vEdge router) matches.

You can display the installed vSmart authorized serial numbers using the **show control valid-vsmarts** command on a vSmart controller or a vEdge router and the **show orchestrator valid-vsmarts** command on a vBond orchestrator. You can display the installed vEdge authorized serial and chassis number associations using the **show control valid-vedges** command on a vSmart controller and the **show orchestrator valid-devices** command on a vBond orchestrator

Now, let's look at how the PKI authentication components and the device serial and chassis numbers are used to authenticate devices on the Viptela overlay network. When vSmart controllers, vBond orchestrators, and vEdge routers first boot up, they establish secure DTLS or TLS connections between them. Over these connections, the devices authenticate each other, using the public and private keys, the signed certificates, and the device serial numbers and performing a series of handshake operations to ensure that all the devices on the network are valid and not imposters. The following figure illustrates the key and certificate exchange that occurs when the Viptela devices boot. For details about the authentication that occurs during the bringup process, see Bringup Sequence of Events .



Control Plane Encryption

Control plane encryption is done by either DTLS, which is based on the TLS protocol, or TLS. These protocol encrypt the control plane traffic that is sent across the connections between Viptela devices to validate the integrity of the data. TLS uses asymmetric cryptography for authenticating key exchange, symmetric encryption for confidentiality, and message authentication codes for message integrity.

A single Viptela device can have DTLS or TLS connections to multiple Viptela devices, so vdaemon creates a kernel route for each destination. For example, a vEdge router would typically have one kernel route, and hence one DTLS or TLS connection, for each vSmart controller. Similarly, a vSmart controller would have one kernel route and one DTLS or TLS connection for each vEdge router in its domain.

Control Plane Integrity

The Viptela design implements control plane integrity by combining two security elements: SHA-1 or SHA-2 message digests, and public and private keys.

SHA-1 and SHA-2 are cryptographic hash functions that generate message digests (sometimes called simply digests) for each packet sent over a control plane connection. SHA-1 generates a 160-bit message digest. SHA-2 is a family that consists of six hash functions with digests that are 224, 256, 384, or 512 bits. The receiver then generates a digest for the packet, and if the two match, the packet is accepted as valid. Both SHA-1 and SHA-2 allow verification that the packet's contents have not been tampered with.

The second component of control plane integrity is the use of public and private keys. When a control plane connection is being established, a local Viptela device sends a challenge to a remote device. The remote device encrypts the challenge by signing it with its private key, and returns the signed challenge to the local device. The local device then uses the remote device's public key to verify that the received challenge matches the sent challenge.

Then, once a control plane connection is up, keys are used to ensure that packets have been sent by a trusted host and were not inserted midstream by an untrusted source. The authenticity of each packet is verified through encryption and decryption with symmetric keys that were exchanged during the process of establishing the control connection.

Additional Information

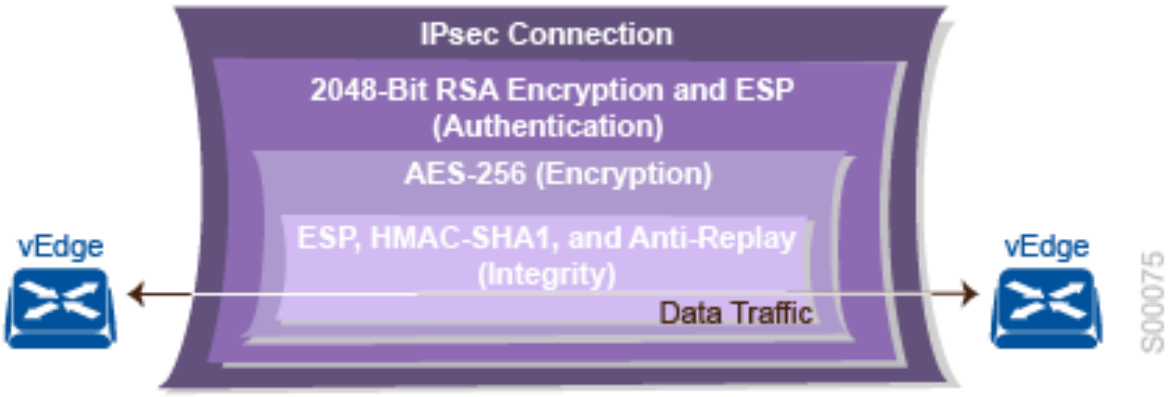
Bringup Sequence of Events
Configuring Security Parameters
Data Plane Security Overview
Security Overview

Data Plane Security Overview

The data plane of any network is responsible for handling data packets that are transported across the network. (The data plane is also sometimes called the forwarding plane.) In a traditional network, data packets are typically sent directly over the Internet or another type of public IP cloud, or they could be sent through MPLS tunnels. If the vEdge routers in the Viptela overlay network were to send traffic over a public IP cloud, the transmission would be insecure. Anyone would be able to sniff the traffic, and it would be easy to implement various types of attacks, including man-in-the-middle (MITM) attacks.

The underlying foundation for security in the Viptela data plane is the security of the control plane. Because the control plane is secure—all devices are validated, and control traffic is encrypted and cannot be tampered with—we can be confident in using routes and other information learned from the control plane to create and maintain secure data paths throughout a network of vEdge routers.

The data plane provides the infrastructure for sending data traffic among the vEdge routers in the Viptela overlay network. Data plane traffic travels within secure Internet Security (IPsec) connections. The Viptela data plane implements the key security components of authentication, encryption, and integrity in the following ways:



- **Authentication** —As mentioned above, the Viptela control plane contributes the underlying infrastructure for data plane security. In addition, authentication is enforced by two other mechanisms:

  - RSA encryption with 2048-bit keys.
  - Two standard protocols from the IPsec security suite framework, Encapsulation Security Payload (ESP) and Authentication Header (AH), are used to authentication the origin of data traffic.

- **Encryption** —The standard ESP protocol protects the data packet's payload, and the standard AH protocol protects both the payload and the non-mutable header fields. Key exchange encryption is done using the AES-256 cipher.
- **Integrity** —To guarantee that data traffic is transmitted across the network without being tampered with, the data plane implements several mechanisms from the IPsec security protocol suite:
  - The ESP protocol encapsulates the payload of data packets.
  - The HMAC-SHA1 algorithm, which is used by the IPsec AH protocol, combines a keyed-hash authentication code with SHA-1 cryptography to ensure data integrity. AH encapsulates the non-mutable fields in the outer IP header and the payload of data packets. You can configure the integrity methods supported on each vEdge router, and this information is exchanged in the router's TLOC properties. If two vEdge peers advertise different authentication types, they negotiate the type to use, choosing the strongest method.
  - The anti-replay scheme protects against attacks in which an attacker duplicates encrypted packets.

Data Plane Authentication and Encryption

Before a pair of vEdge routers can exchange data traffic, they establish an IPsec connection between them, which they use as a secure communications channel, and then the routers authenticate each other over this connection. As with the control plane, the data plane uses keys to perform Viptela device authentication.

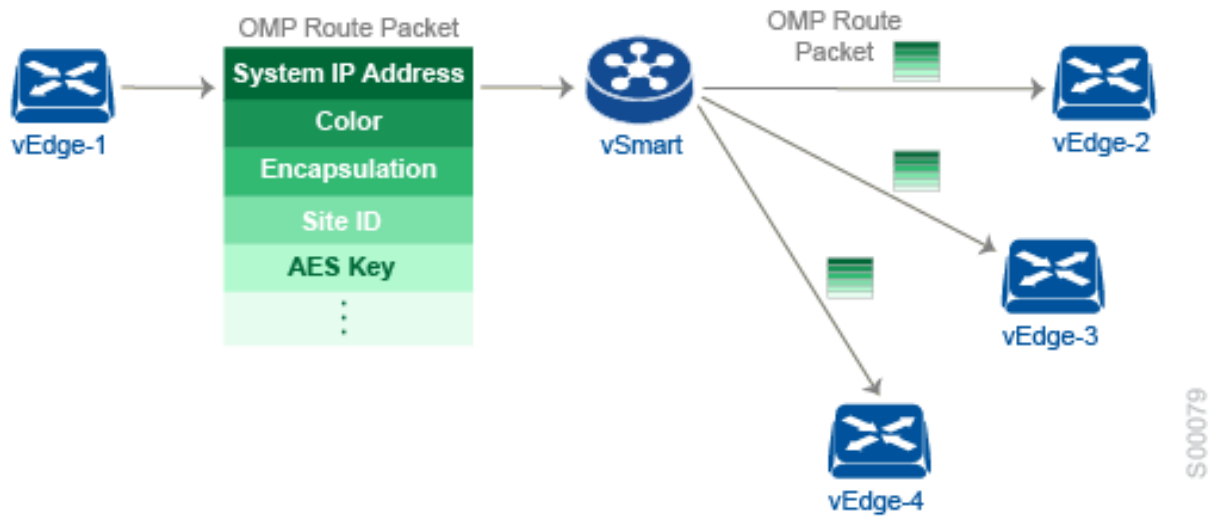
In a traditional IPsec environment, key exchange is handled by the Internet Key Exchange (IKE) protocol. IKE first sets up secure communications channels between devices and then establishes security associations (SAs) between each pair of devices that want to exchange data. IKE uses a Diffie-Hellman key exchange algorithm to generate a shared key that encrypts further IKE communication. To establish SAs, each device (n) exchanges keys with every other device in the network and creates per-pair keys, generating a unique key for each remote device. This scheme means that in a fully meshed network, each device has to manage n_2 key exchanges and (n-1) keys. As an example, in a 1,000-node network, 1,000,000 key exchanges are required to authenticate the devices, and each node is responsible for maintaining and managing 999 keys.

The discussion in the previous paragraph points out why an IKE-style key exchange does not scale as network size increases and why IKE could be a bottleneck in starting and in maintaining data exchange on a large network:

- The handshaking required to set up the communications channels is both time consuming and resource intensive.
- The processing required for the key exchange, especially in larger networks, can strain network resources and can take a long time.

The Viptela implementation of data plane authentication and encryption establishes SAs between each pair of devices that want to exchange data, but it dispenses with IKE altogether. Instead, to provide a scalable solution to data plane key exchange, the Viptela solution takes advantage of the fact that the DTLS control plane connections in the Viptela overlay network are known to be secure. Because the Viptela control plane establishes authenticated, encrypted, and tamperproof connections, there is no need in the data plane to set up secure communications channels to perform data plane authentication.

In the Viptela network, data plane encryption and key generation are done by AES-256, a symmetric-key algorithm that uses the same key to encrypt outgoing packets and to decrypt incoming packets. Each vEdge router periodically generates an AES key for its data path (specifically, one key per TLOC) and transmits this key to the vSmart controller in OMP route packets, which are similar to IP route updates. These packets contain information that the vSmart controller uses to determine the network topology, including the vEdge router's TLOC (a tuple of the system IP address and traffic color) and AES key. The vSmart controller then places these OMP route packets into reachability advertisements that it sends to the other vEdge routers in the network. In this way, the AES keys for all the vEdge routers are distributed across the network. Even though the key exchange is symmetric, Viptela devices use it in an asymmetric fashion. The result is a simple and scalable key exchange process that does not use per-pair keys.



If control policies configured on a vSmart controller limit the communications channels between network devices, the reachability advertisements sent by the vSmart controller contain information only for the vEdge routers that they are allowed to exchange data with. So, a vEdge router learns the keys only for those vEdge routers that they are allowed to communicate with.

To further strengthen data plane authentication and encryption, vEdge routers regenerate their AES keys aggressively (by default, every 24 hours). Also, the key regeneration mechanism ensures that no data traffic is dropped when keys change.

In the Viptela overlay network, the liveness of SAs between vEdge router peers is tracked by monitoring BFD packets, which are periodically exchanged over the IPsec connection between the peers. IPsec relays the connection status to the vSmart controllers. If data connectivity between two peers is lost, the exchange of BFD packets stops, and from this, the vSmart controller learns that the connection has been lost.

The Viptela IPsec software has no explicit SA idle timeout, which specifies the time to wait before deleting SAs associated with inactive peers. Instead, an SA remains active as long as the IPsec connection between two vEdge routers is up, as determined by the periodic exchange of BFD packets between them. Also, the frequency with which SA keys are regenerated obviates the need to implement an implicit SA idle timeout.

In summary, the Viptela data plane authentication offers the following improvements over IKE:

- Because only n +1 keypaths are required rather than the n $_2$ required by IKE, the Viptela solution scales better as the network grows large.
- Keys are generated and refreshed locally, and key exchange is performed over a secure control plane.
- No key server is required, and thus there is no need to worry about high availability requirements of a key server.

Data Plane Integrity

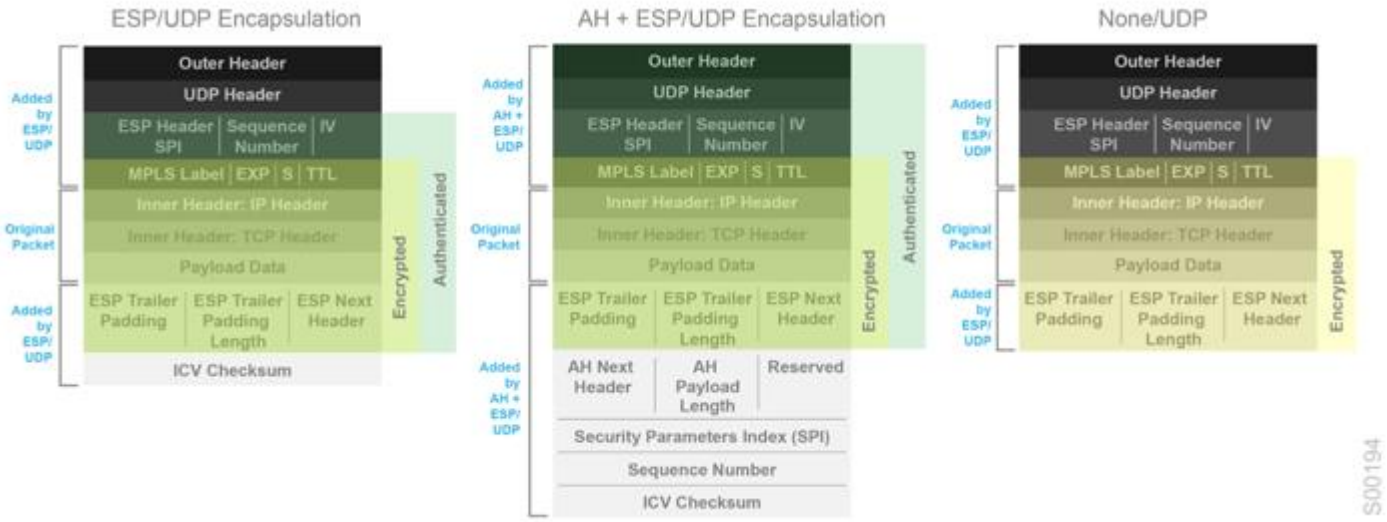
A number of components contribute to the integrity of data packets in the Viptela data plane:

- ESP, which is the standard IPsec encryption protocol, protects (via encryption and authentication) the inner header, data packet payload, and ESP trailer in all data packets.
- AH, which is the standard IPsec authentication protocol, protects (via authentication) the entire data packet, including the inner and outer headers, data packet payload, and ESP trailer.
- Anti-replay, which is also part of the standard IPsec software suite, provides a mechanism to number all data packets and to ensure that receiving routers accept only packets with unique numbers.

The first of these components, ESP, is the standard IPsec encryption protocol. ESP protects a data packet's payload and its inner IP header fields both by encryption, which occurs automatically, and authentication. For authentication, ESP performs a checksum calculation on the data packet's payload and inner header fields and places the resultant hash (also called a digest) into a 12-byte HMAC-SHA1 field at the end of the packet. (A hash is a one-way compression.) The receiving device performs the same checksum and compares its calculated hash with that in the packet. If the two checksums match, the packet is accepted. Otherwise, it is dropped. In the figure below, the left stack illustrates the ESP/UDP encapsulation. ESP encrypts and authenticates the inner headers, payload, MPLS label (if present), and ESP trailer fields, placing the HMAC-SHA1 hash in the ICV checksum field at the end of the packet. The outer header fields added by ESP/UDP are neither encrypted nor authenticated.

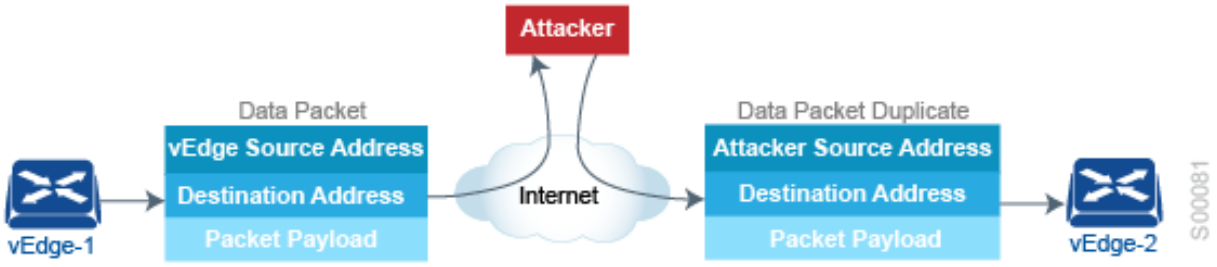
A second component that contributes to data packet integrity is AH, the standard IPsec authentication protocol, which protects all fields in a data packet via authentication. AH performs a checksum process similar to that done by ESP, except that instead of calculating the checksum over just the payload and inner IP header fields, it calculates it over all the fields in the packet—the payload, the inner header, and all the non-mutable fields in the outer IP header. AH places the resultant HMAC-SHA1 hash into the last field of the packet. As with ESP, AH on the receiving device performs the same checksum, and accepts packets whose checksums match. In the figure below, the center stack illustrates the encapsulation performed by AH, in combination with ESP. ESP again encrypts the inner headers, payload, MPLS label (if present), and ESP trailer fields, and now AH authenticates the entire packet—the outer IP and UDP headers, the ESP header, the MPLS label (if present), the original packet, and the ESP trailer—and places its calculated HMAC-SHA1 hash into the ICV checksum field at the end of the packet.

For situations in which data packet authentication is not required, you can disable data packet authentication altogether. In this case, data packets are processed just by ESP, which encrypts the original packet, the MPLS label (if present), and the ESP trailer. This scheme is illustrated in the right stack in the figure below.

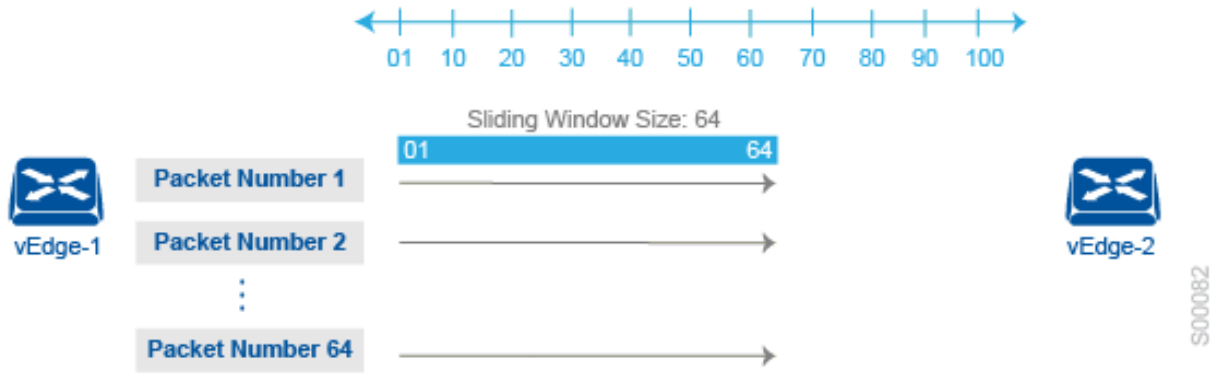


Note that Viptela devices exchange not only the encryption key (which is symmetric), but also the authentication key that is used to generate the HMAC-SHA1 digest. Both are distributed as part of the TLOC properties for a vEdge router.

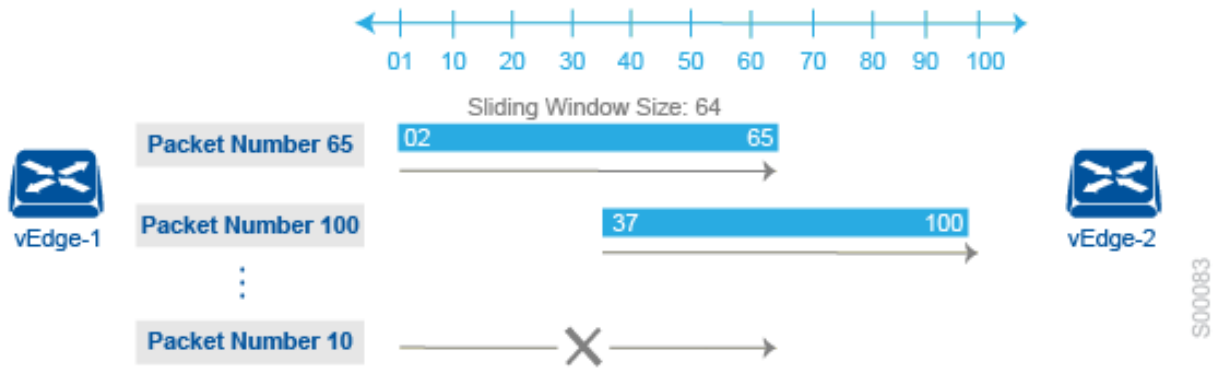
Even though the IPsec connections over which data traffic is exchanged are secure, they often travel across a public network space, such as the Internet, where it is possible for a hacker to launch a replay attack (also called a man-in-the-middle, or MITM, attack) against the IPsec connection. In this type of attack, an adversary tampers with the data traffic by inserting a copy of a message that was previously sent by the source. If the destination cannot distinguish the replayed message from a valid message, it may authenticate the adversary as the source or may incorrectly grant to the adversary unauthorized access to resources or services.



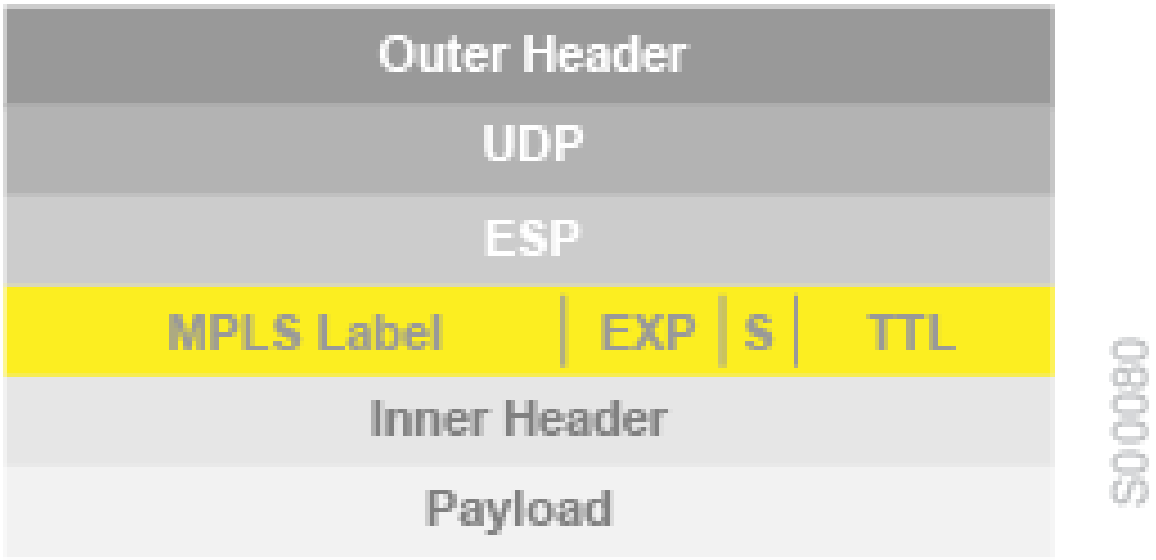
As a counter to such attacks, the Viptela overlay network software implements the IPsec anti-replay protocol. This protocol consists of two components, both of which protect the integrity of a data traffic stream. The first component is to associate sequence numbers with each data packets. The sender inserts a sequence number into each IPsec packet, and the destination checks the sequence number, accepting only packets with unique, non-duplicate sequence numbers. The second component is a sliding window, which defines a range of sequence numbers that are current. The sliding window has a fixed length. The destination accepts only packets whose sequence numbers fall within the current range of values in the sliding window, and it drops all others. A sliding window is used rather than accepting only packets whose sequence number is larger than the last known sequence number, because packets often do not arrive in order.



When the destination receives a packet whose sequence number is larger than the highest number in the sliding window, it slides the window to the right, thus changing the range of valid sequences numbers it will accept. This scheme protects against an MITM type of attack because, by choosing the proper window size, you can ensure that if a duplicate packet is inserted into the traffic stream, its sequence number will either be within the current range but will be a duplicate, or it will be smaller than the lowest current value of the sliding window. Either way, the destination will drop the duplicate packet. So, the sequence numbering combined with a sliding window provide protection against MITM type of attacks and ensure the integrity of the data stream flowing within the IPsec connection.



Carrying VPN Information in Data Packets



For enterprise-wide VPNs, Viptela devices support MPLS extensions to data packets that are transported within IPsec connections. The figure to the right shows the location of the MPLS information in the data packet header. These extensions provide the security for the network segmentation (that is, for the VPNs) that is needed to support multi-tenancy in a branch or segmentation in a campus. The Viptela implementation uses IPsec UDP-based overlay network layer protocol encapsulation as defined in RFC 4023 . The security is provided by including the Initialization Vector (IV) at the beginning of the payload data in the ESP header. The IV value is calculated by the AES-256 cipher block chaining (CBC).

Additional Information

Configuring Security Parameters
Control Plane Security Overview
Security Overview

Configuring Security Parameters

This article describes how to change security parameters for the control plane and the data plane in the Viptela overlay network.

Configure Control Plane Security Parameters

By default, the control plane uses DTLS as the protocol that provides privacy on all its tunnels. DTLS runs over UDP.

You can change the control plane security protocol to TLS, which runs over TCP. The primary reason to use TLS is that, if you consider the vSmart controller to be a server, firewalls protect TCP servers better than UDP servers.

You configure the control plane tunnel protocol on a vSmart controller:

```
vSmart(config)# security control protocol tls
```

With this change, all control plane tunnels between the vSmart controller and vEdge routers and between the controller and vManage NMSs use TLS. Control plane tunnels to vBond orchestrators always use DTLS, because these connections must be handled by UDP.

In a domain with multiple vSmart controllers, when you configure TLS on one of the vSmart controllers, all control plane tunnels from that controller to the other controllers use TLS. Said another way, TLS always takes precedence over DTLS. However, from the perspective of the other vSmart controllers, if you have not configured TLS on them, they use TLS on the control plane tunnel only to that one vSmart controller, and they use DTLS tunnels to all the other vSmart controllers and to all their connected vEdge routers. To have all vSmart controllers use TLS, configure it on all of them.

By default, the vSmart controller listens on port 23456 for TLS requests. To change this:

```
vSmart(config)# security control tls-port number
```

The port can be a number from 1025 through 65535.

To display control plane security information, use the **show control connections** command on the vSmart controller. For example:

```
vSmart-2# show control connections
                                                                                    PEER
PEER
PEER     PEER     PEER             SITE        DOMAIN      PEER             PRIVATE  PEER
PUBLIC
TYPE     PROTOCOL SYSTEM IP        ID          ID          PRIVATE IP       PORT     PUBLIC IP
PORT     REMOTE COLOR     STATE           UPTIME
-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------
vedge    dtls     172.16.255.11    100         1           10.0.5.11        12346    10.0.5.11
12346    lte              up                  0:07:48:58
vedge    dtls     172.16.255.21    100         1           10.0.5.21        12346    10.0.5.21
12346    lte              up                  0:07:48:51
vedge    dtls     172.16.255.14    400         1           10.1.14.14       12360    10.1.14.14
12360    lte              up                  0:07:49:02
vedge    dtls     172.16.255.15    500         1           10.1.15.15       12346    10.1.15.15
12346    default          up                  0:07:47:18
vedge    dtls     172.16.255.16    600         1           10.1.16.16       12346    10.1.16.16
12346    default          up                  0:07:41:52
vsmart   tls      172.16.255.19    100         1           10.0.5.19        12345    10.0.5.19
12345    default          up                  0:00:01:44
vbond    dtls     -                0           0           10.1.14.14       12346    10.1.14.14
12346    default          up                  0:07:49:08

vSmart-2 control connections
                                                                                    PEER
PEER
PEER     PEER     PEER             SITE        DOMAIN      PEER             PRIVATE  PEER
PUBLIC
TYPE     PROTOCOL SYSTEM IP        ID          ID          PRIVATE IP       PORT     PUBLIC IP
PORT     REMOTE COLOR     STATE           UPTIME
-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------
vedge    tls      172.16.255.11    100         1           10.0.5.11        12345    10.0.5.11
12345    lte              up                  0:00:01:18
vedge    tls      172.16.255.21    100         1           10.0.5.21        12345    10.0.5.21
12345    lte              up                  0:00:01:18
vedge    tls      172.16.255.14    400         1           10.1.14.14       12345    10.1.14.14
12345    lte              up                  0:00:01:18
vedge    tls      172.16.255.15    500         1           10.1.15.15       12345    10.1.15.15
12345    default          up                  0:00:01:18
vedge    tls      172.16.255.16    600         1           10.1.16.16       12345    10.1.16.16
12345    default          up                  0:00:01:18
vsmart   tls      172.16.255.20    200         1           10.0.12.20       23456    10.0.12.20
23456    default          up                  0:00:01:32
vbond    dtls     -                0           0           10.1.14.14       12346    10.1.14.14
12346    default          up                  0:00:01:33
```

Configure DTLS on vManage NMS

If you configure the vManage NMS to use TLS as the control plane security protocol, you must enable port forwarding on your NAT. If you are using DTLS as the control plane security protocol, you do not need to do anything.

The number of ports forwarded depends on the number of vdaemon processes running on the vManage NMS. To display information about these processes and about and the number of ports that are being forwarded, use the **show control summary** command shows that four vdaemon processes are running:

```
vManage# show control summary
          VBOND     VMANAGE    VSMART    VEDGE
INSTANCE  COUNTS    COUNTS     COUNTS    COUNTS
-----------------------------------------------
0           2          0          2        7
1           2          0          0        5
2           2          0          0        5
3           2          0          0        4
```

To see the listening ports, use the **show control local-properties** command:

```
vManage# show control local-properties
organization-name            vIPtela Inc Test
certificate-status           Installed
root-ca-chain-status         Installed

certificate-validity         Valid
certificate-not-valid-before May 20 00:00:00 2015 GMT
certificate-not-valid-after  May 20 23:59:59 2016 GMT

dns-name                     vbond.viptela.com
site-id                      5000
domain-id                    0
protocol                     dtls
tls-port                     23456
...
...
...
number-active-wan-interfaces 1

                  PUBLIC         PUBLIC  PRIVATE        PRIVATE
ADMIN     OPERATION  LAST
INDEX  INTERFACE IP             PORT    IP             PORT     VSMARTS  VMANAGES  COLOR
CARRIER           STATE     STATE      CONNECTION
-------------------------------------------------------------------------------------------------------
-----------------------------------------------------
0      eth0      72.28.108.37   12361   172.16.98.150  12361    2        0         silver
default          up        up        0:00:00:08
```

This output shows that the listening TCP port is 23456. If you are running vManage NMS behind a NAT, you should open the following ports on the NAT device:

- 23456 (base - instance 0 port)
- 23456 + 100 (base + 100)
- 23456 + 200 (base + 200)
- 23456 + 300 (base + 300)

Note that the number of instances is the same as the number of cores you have assigned for the vManage NMS, up to a maximum of 8.

Configure Data Plane Security Parameters

In the data plane, IPsec is enabled by default on all vEdge routers, and by default IPsec tunnel connections use the AH-SHA1 HMAC for authentication on the IPsec tunnels. On vEdge routers, you can change the type of authentication, and you can modify the IPsec rekeying timer and the size of the IPsec anti-replay window.

Configure Allowed Authentication Types

By default, IPsec tunnel connections use AH-SHA1 HMAC and ESP HMAC-SHA1 for authentication, choosing whichever authentication method is stronger. To modify the negotiated authentication types or to disable authentication, use the following command:

```
vEdge(config)# security ipsec authentication-type (ah-no-id | ah-sha1-hmac | none | sha1-hmac)
```

Configure each authentication type with a separate **security ipsec authentication-type** command. The command options map to the following authentication types, which are listed in order from most strong to least strong:

- **ah-sha1-hmac** enables AH-SHA1 HMAC and ESP HMAC-SHA1.
- **ah-no-id** enables a modified version of AH-SHA1 HMAC and ESP HMAC-SHA1. This option accommodates some non-Viptela devices, including the Apple AirPort Express NAT, that have a bug that causes the ID field in the IP header, a non-mutable field, to be modified. Configure the **ah-no-id** option in the list of authentication types to have the Viptela AH software ignore the ID field in the IP header so that the Viptela software can work in conjunction with these devices.
- **sha1-hmac** enables ESP HMAC-SHA1.
- **none** maps to no authentication. You can choose this option in situations where data plane authentication and integrity are not a concern.

For information about which data packet fields are affected by these authentication types, see the "Data Plane Integrity" section in Data Plane Security Overview .

vEdge routers advertise their configured authentication types in their TLOC properties. The two routers on either side of an IPsec tunnel connection negotiate the authentication to use on the connection between them, using the strongest authentication type that is configured on both of the routers. For example, if one vEdge router advertises AH-HMAC-SHA1, ESP HMAC-SHA1, and none, and a second vEdge router advertises ESP HMAC-SHA1 and none, the two routers negotiate to use ESP HMAC-SHA1 on the IPsec tunnel connection between them. If no common authentication types are configured on the two vEdge peers, no IPsec tunnel is established between them.

The encryption algorithm on IPsec tunnel connections is either AES-256-GCM or AES-256-CBC. For unicast traffic, if the remote side supports AES-256-GCM, that encryption algorithm is used. Otherwise, AES-256-CBC is used. For multicast traffic, the encryption algorithm is AES-256-CBC. You cannot modify the choice made by the software.

When the IPsec authentication type is changed, the AES key for the data path is changed.

Change the Rekeying Timer

Before vEdge routers can exchange data traffic, they set up a secure authenticated communications channel between them. The vEdge routers use the DTLS or TLS control plane connection between them as the channel, and they use the AES-256 cipher to perform encryption. Each vEdge router generates a new AES key for its data path periodically.

By default, a key is valid for 86400 seconds (24 hours), and the timer range is 10 seconds through 1209600 seconds (14 days). To change the rekey timer value:

```
vEdge(config)# security ipsec rekey seconds
```

The configuration looks like this:

```
security
  ipsec
```

```
    rekey seconds
  !
```

When the IPsec keys are compromised, you can generate new keys immediately, without modifying the configuration of the vEdge router. To do this, issue the **request security ipsec-rekey** command on the compromised vEdge router.

For example, the following output shows that the local SA has a SPI (key) of 256:

```
vEdge# show ipsec local-sa

                                        SOURCE           SOURCE
TLOC ADDRESS     TLOC COLOR       SPI   IP               PORT    KEY HASH
-------------------------------------------------------------------------------
172.16.255.15    lte              256   10.1.15.15       12346   *****b93a
```

If this key is compromised, use the **request security ipsec-rekey** command to generate a new key immediately. This command increments the existing key, so in our example the SPI changes to 257:

```
vEdge# request security ipsec-rekey
vEdge# show ipsec local-sa
                                        SOURCE           SOURCE
TLOC ADDRESS     TLOC COLOR       SPI   IP               PORT    KEY HASH
-------------------------------------------------------------------------------
172.16.255.15    lte              257   10.1.15.15       12346   *****b93a
```

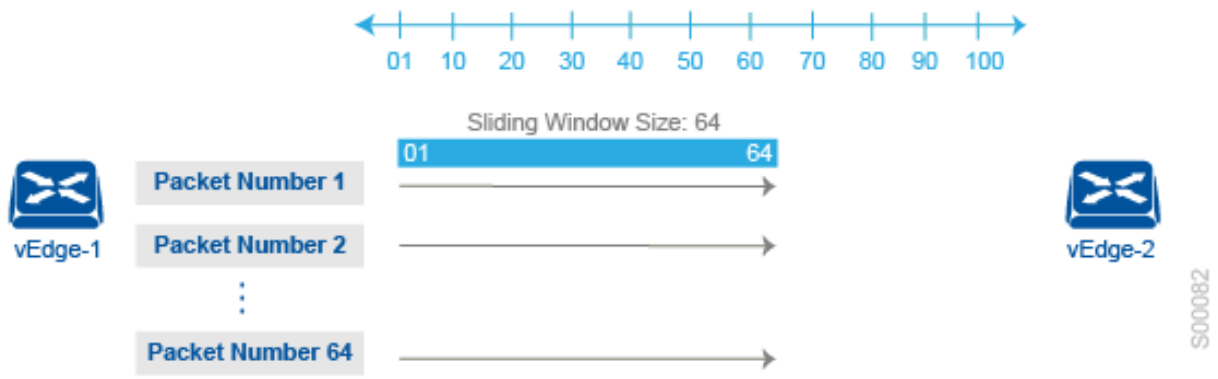
After the new key is generated, the vEdge router sends it immediately to all its DTLS or TLS peers, and they begin using it as soon as they receive it. Note that the old compromised SPI (256) will continue to be used for a short period of time, until it times out.

To stop using the compromised key immediately, issue the **request security ipsec-rekey** command twice, in quick succession. This sequence of commands removes both SPI 256 and 257, and sets the key to 258. Note, however, that some packets will be dropped for a short period of time, until all the remote vEdge routers learn the new key.

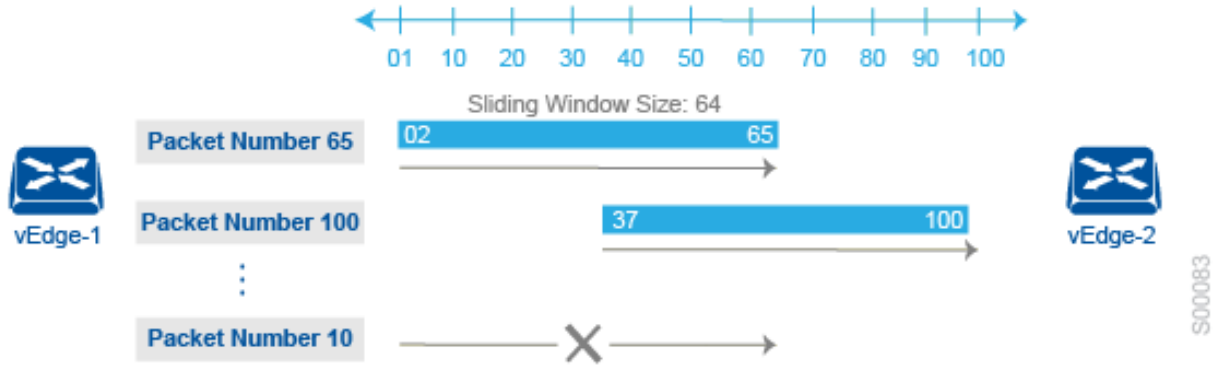
```
vEdge# request security ipsec-rekey
vEdge# request security ipsec-rekey
vEdge# ipsec local-sa
                                        SOURCE           SOURCE
TLOC ADDRESS     TLOC COLOR       SPI   IP               PORT    KEY HASH
-------------------------------------------------------------------------------
172.16.255.15    lte              258   10.1.15.15       12346   *****b93a
```

Change the Size of the Anti-Replay Window

IPsec authentication provides anti-replay protection by assigning a unique sequence number to each packet in a data stream. This sequence numbering protects against an attacker duplicating data packets. With anti-replay protection, the sender assigns monotonically increasing sequence numbers, and the destination checks these sequence numbers to detect duplicates. Because packets often do not arrive in order, the destination maintains a sliding window of sequence numbers that it will accept.



Packets with sequence numbers that fall to the left of the sliding window range are considered old or duplicates, and the destination drops them. The destination tracks the highest sequence number it has received, and adjusts the sliding window when it receives a packet with a higher value.



By default, the sliding window is set to 512 packets. It can be set to any value between 64 and 8192 that is a power of 2 (that is, 64, 128, 256, 512, 1024, 2048, 4096, or 8192). To modify the anti-replay window size, use the **replay-window** command, specifying the size of the window:

```
vEdge(config)# security ipsec replay-window  number
```

The configuration looks like this:

```
security
  ipsec
    replay-window number
  !
!
```

If QoS is configured on a vEdge router, that router might experience a larger than expected number of packet drops as a result of the IPsec anti-replay mechanism, and many of the packets that are dropped are legitimate ones. This occurs because QoS reorders packets, giving higher-priority packets preferential treatment and delaying lower-priority packets. To minimize or prevent this situation, increase the size of the anti-replay window.

Additional Information

show control connections
show security-info
Security Overview

Configuring IKE-Enabled IPsec Tunnels

To securely transfer traffic from the Viptela overlay network to a service network, you can configure IPsec tunnels that run the Internet Key Exchange (IKE) protocol. IKE-enabled IPsec tunnels provide authentication and encryption to ensure secure packet transport.

You create an IKE-enabled IPsec tunnel by configuring an IPsec interface. IPsec interfaces are logical interfaces, and you configure them just like any other physical interface. You configure IKE protocol parameters on the IPsec interface, and you can configure other interface properties.

The Viptela software supports IKE, Version 1, as defined in RFC 2409 , *Internet Key Exchange* , and IKE, Version 2, as defined in RFC 7296 , *Internet Key Exchange Protocol, Version 2* .

One use for IPsec tunnels is to allow vEdge Cloud router VM instances running on Amazon AWS to connect to the Amazon virtual private cloud (VPC). You must configure IKE Version 1 on these routers.

Configure an IPsec Tunnel

To configure an IPsec tunnel interface for secure transport traffic from a service network, you create a logical IPsec interface:

```
vEdge(config)# vpn vpn-id interface ipsecnumber
vEdge(config-interface-ipsec)# ip address ipv4-prefix/length
vEdge(config-interface-ipsec)# (tunnel-source ip-address | tunnel-source-interface interface-name)
vEdge(config-interface-ipsec)# tunnel-destination ipv4-address
vEdge(config-interface-ipsec)# no shutdown
```

You can create the IPsec tunnel in any service VPN (VPN 1 through 65530, omitting 512).

The IPsec interface has a name in the format **ipsec** *number* , where *number* can be from 1 through 255.

Each IPsec interface must have an IPv4 address. This address must be a /30 prefix. All traffic in the VPN that is within this IPv4 prefix is directed to a physical interface in VPN 0 to be sent securely over an IPsec tunnel.

To configure the source of the IPsec tunnel on the local device, you can specify either the IP address of the physical interface (in the **tunnel-source** command) or the name of the physical interface (in the **tunnel-source-interface** command). Ensure that the physical interface is configured in VPN 0.

To configure the destination of the IPsec tunnel, specify the IP address of the remote device in the **tunnel-destination** command.

The combination of a source address (or source interface name) and a destination address defines a single IPsec tunnel. Only one IPsec tunnel can exist that uses a specific source address (or interface name) and destination address pair.

Enable IKE Version 1

When you create an IPsec tunnel on a vEdge router, IKE Version 1 is enabled by default on the tunnel interface. The following properties are also enabled by default for IKEv1:

- Authentication and encryption—AES-256 advanced encryption standard CBC encryption with the HMAC-SHA1 keyed-hash message authentication code algorithm for integrity
- Diffie-Hellman group number—16
- Rekeying time interval—4 hours
- SA establishment mode—Main

By default, IKEv1 uses IKE main mode to establish IKE SAs. In this mode, six negotiation packets are exchanged to establish the SA. To exchange only three negotiation packets, enable aggressive mode:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# mode aggressive
```

By default, IKEv1 uses Diffie-Hellman group 16 in the IKE key exchange. This group uses the 4096-bit more modular exponential (MODP) group during IKE key exchange. You can change the group number to 2 (for 1024-bit MODP), 14 (2048-bit MODP), or 15 (3072-bit MODP):

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# group number
```

By default, IKE key exchange uses AES-256 advanced encryption standard CBC encryption with the HMAC-SHA1 keyed-hash message authentication code algorithm for integrity. You can change the authentication to AES-128 CBC and the integrity checking to the HMAC-SHA1 keyed-has message authentication code algorithm:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# cipher-suite aes128-cbc-sha1
```

By default, IKE keys are refreshed every 1 hours (3600 seconds). You can change the rekeying interval to a value from 30 seconds through 14 days (1209600 seconds). It is recommended that the rekeying interval be at least 1 hour.

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# rekey seconds
```

To force the generation of new keys for an IKE session, issue the **request ipsec ike-rekey** command.

vEdge(config)# **vpn** *vpn-id* **interface ipsec** *number* **ike**

For IKE, you can also configure preshared key (PSK) authentication:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# authentication-type pre-shared-key pre-shared-secret password
```

password is the password to use with the preshared key. It can be an ASCII or a hexadecimal string from 1 through 127 characters long.

If the remote IKE peer requires a local or remote ID, you can configure this identifier:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike authentication-type
vEdge(config-authentication-type)# local-id id
vEdge(config-authentication-type)# remote-id id
```

The identifier can be an IP address or any text string from 1 through 63 characters long. By default, the local ID is the tunnel's source IP address and the remote ID is the tunnel's destination IP address.

Enable IKE Version 2

When you configure an IPsec tunnel to use IKE Version 2, the following properties are also enabled by default for IKEv2:

- Authentication and encryption—AES-256 advanced encryption standard CBC encryption with the HMAC-SHA1 keyed-hash message authentication code algorithm for integrity
- Diffie-Hellman group number—16
- Rekeying time interval—4 hours

By default, IKEv2 uses Diffie-Hellman group 16 in the IKE key exchange. This group uses the 4096-bit more modular exponential (MODP) group during IKE key exchange. You can change the group number to 2 (for 1024-bit MODP), 14 (2048-bit MODP), or 15 (3072-bit MODP):

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# group number
```

By default, IKE key exchange uses AES-256 advanced encryption standard CBC encryption with the HMAC-SHA1 keyed-hash message authentication code algorithm for integrity. You can change the authentication to AES-128 CBC and the integrity checking to the HMAC-SHA1 keyed-has message authentication code algorithm:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# cipher-suite aes128-cbc-sha1
```

By default, IKE keys are refreshed every 4 hours (14,400 seconds). You can change the rekeying interval to a value from 30 seconds through 14 days (1209600 seconds):

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# rekey seconds
```

To force the generation of new keys for an IKE session, issue the **request ipsec ike-rekey** command.

For IKE, you can also configure preshared key (PSK) authentication:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike
vEdge(config-ike)# authentication-type pre-shared-key pre-shared-secret password
```

password is the password to use with the preshared key. It can be an ASCII or a hexadecimal string, or it can be an AES-encrypted key.

If the remote IKE peer requires a local or remote ID, you can configure this identifier:

```
vEdge(config)# vpn vpn-id interface ipsecnumber ike authentication-type
vEdge(config-authentication-type)# local-id id
vEdge(config-authentication-type)# remote-id id
```

The identifier can be an IP address or any text string from 1 through 64 characters long. By default, the local ID is the tunnel's source IP address and the remote ID is the tunnel's destination IP address.

Configure IPsec Tunnel Parameters

By default, the following parameters are used on the IPsec tunnel that carries IKE traffic:

- Authentication and encryption—AES-256 algorithm in GCM (Galois/counter mode)
- Rekeying interval—4 hours
- Replay window—32 packets

You can change the encryption on the IPsec tunnel to the AES-256 cipher in CBC (cipher block chaining mode, with HMAC-SHA1-96 keyed-hash message authentication or to null, to not encrypt the IPsec tunnel used for IKE key exchange traffic:

```
vEdge(config-interface-ipsecnumber)# ipsec
vEdge(config-ipsec)# cipher-suite (aes256-cbc-sha1 | aes256-gcm | null-sha1)
```

By default, IKE keys are refreshed every 4 hours (14,400 seconds). You can change the rekeying interval to a value from 30 seconds through 14 days (1209600 seconds):

```
vEdge(config-interface-ipsecnumber)# ipsec
vEdge(config-ipsec)# rekey seconds
```

To force the generation of new keys for an IPsec tunnel, issue the **request ipsec ipsec-rekey** command.

By default, perfect forward secrecy (PFS) is enabled on IPsec tunnels, to ensure that past sessions are not affected if future keys are compromised. PFS forces a new Diffie-Hellman key exchange, by default using the 4096-bit Diffie-Hellman prime module group. You can change the PFS setting:

```
vEdge(config-interface-ipsecnumber)# ipsec
vEdge(config-ipsec)# perfect-forward-secrecy pfs-setting
```

pfs-setting can be one of the following:

- **group-2** —Use the 1024-bit Diffie-Hellman prime modulus group.
- **group-14** —Use the 2048-bit Diffie-Hellman prime modulus group.
- **group-15** —Use the 3072-bit Diffie-Hellman prime modulus group.

- **group-16** —Use the 4096-bit Diffie-Hellman prime modulus group. This is the default.
- **none** —Disable PFS.

By default, the IPsec replay window on the IPsec tunnel is 512 bytes. You can set the replay window size to 64, 128, 256, 512, 1024, 2048, or 4096 packets:

```
vEdge(config-interface-ipsecnumber)# ipsec
vEdge(config-ipsec)# replay-window number
```

Modify IKE Dead-Peer Detection

IKE uses a dead-peer detection mechanism to determine whether the connection to an IKE peer is functional and reachable. To implement this mechanism, IKE sends a Hello packet to its peer, and the peer sends an acknowledgment in response. By default, IKE sends Hello packets every 10 seconds, and after three unacknowledged packets, IKE declares the neighbor to be dead and tears down the tunnel to the peer. Thereafter, IKE periodically sends a Hello packet to the peer, and re-establishes the tunnel when the peer comes back online.

You can change the liveness detection interval to a value from 0 through 65535 seconds, and you change change the number of retries to a value from 0 through 255:

```
vEdge(config-interface-ipsecnumber)# dead-peer-detection seconds retries number
```

Configure Other Interface Properties

For IPsec tunnel interfaces, you can configure only the following additional interface properties:

```
vEdge(config-interface-ipsec)# mtu bytes
vEdge(config-interface-ipsec)# tcp-mss-adjust bytes
```

Additional Information

Security Overview

Security CLI Reference

CLI commands for configuring and monitoring security.

Security Configuration Commands

Use the following commands to configure security parameters:

```
security
  control
    protocol (dtls | tls)
    tls-port number
  ipsec
    authentication-type type
    rekey seconds
    replay-window number
vpn vpn-id
  interface ipsecnumber
    access-list acl-name
    block-non-source-ip
    clear-dont-fragment
    dead-peer-detection interval seconds retries number
    description text
    ike
```

```
    authentication-type type
      local-id id
      pre-shared-secret password
      remote-id id
    cipher-suite suite
    group number
    mode mode
    rekey seconds
    version number
  ip address ipv4-prefix/length
  ipsec
    cipher-suite suite
    perfect-forward-secrecy pfs-setting
    rekey seconds
    replay-window number
  mtu bytes
  policer policer-name
  rewrite-rule rule-name
  [no] shutdown
  tcp-mss-adjust bytes
  tunnel-destination (dns-name | ipv4-address)
  (tunnel-source ip-address | tunnel-source-interface interface-name)
```

Security Monitoring Commands

- show control connections
- show security-info

Additional Information

- Configuring IKE-Enabled IPsec Tunnels
- Configuring Security Parameters
- Security Overview

Policy Basics

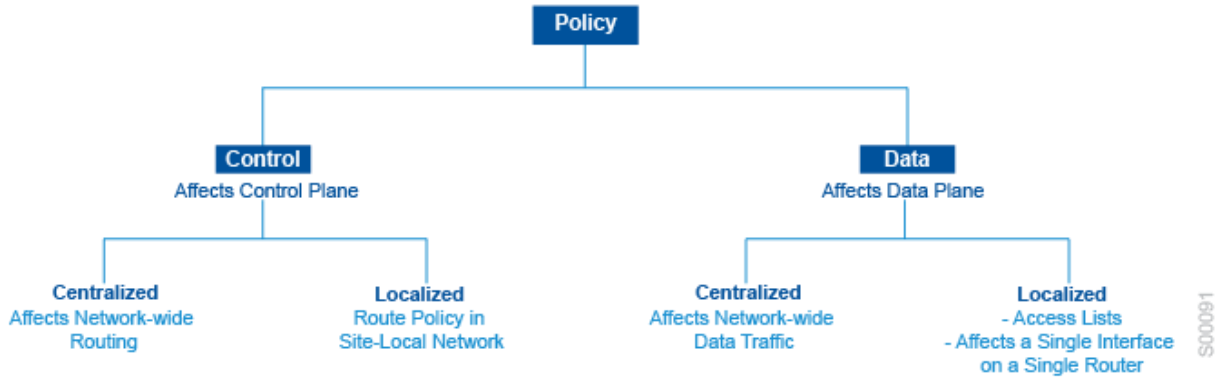
Policy Overview

Policy is used to influence the flow of data traffic among the vEdge routers in the overlay network. Policy comprises:

- *Routing policy* , which affects the flow of routing information in the network's control plane
- *Data policy* , which affects the flow of data traffic in the network's data plane

To implement enterprise-specific traffic control requirements, you create basic policies, and you deploy advanced features of the Viptela software that are activated by means of the policy configuration infrastructure.

Just as the Viptela overlay network architecture clearly separates the control plane from the data plane and clearly separates control between centralized and localized functions, the Viptela policy software is cleanly separated: policies apply either to control plane or data plane traffic, and they are configured either centrally (on vSmart controllers) or locally (on vEdge routers). The following figure illustrates the division between control and data policy, and between centralized and local policy.



The design of the Viptela policy software distinguishes between basic and advanced policy. Basic policy allows you to influence or determine basic traffic flow through the overlay network. Here, you perform standard policy tasks, such as managing the paths along which traffic is routed through the network, and permitting or blocking traffic based on the address, port, and DSCP fields in the packet's IP header. You can also control the flow of data traffic into and out of a vEdge router's interfaces, enabling features such as class of service and queuing, mirroring, and policing.

Advanced features of Viptela policy software offer specialized policy-based network applications. Examples of these applications include the following:

- Service chaining, which redirects data traffic to shared devices in the network, such as firewall, intrusion detection and prevention (IDS), load balancer, and other devices, before the traffic is delivered to its destination. Service chaining obviates the need to have a separate device at each branch site.
- Application-aware routing, which selects the best path for traffic based on real-time network and path performance characteristics.
- Cflowd, for monitoring traffic flow.
- Converting a vEdge router into a NAT device, to allow traffic destined for the Internet or other public network can exit directly from the vEdge router.

By default, no policy of any kind is configured on Viptela devices, either on the centralized vSmart controllers or the local vEdge routers. When control plane traffic, which distributes route information, is unpolicied:

- All route information that OMP propagates among the Viptela devices is shared, unmodified, among all vSmart controllers and all vEdge routers in the overlay network domain.
- No BGP or OSPF route policies are in place to affect the route information that vEdge routers propagate within their local site network.

When data plane traffic is unpolicied, all data traffic is directed towards its destination based on solely on the entries in the local vEdge router's route table, and all VPNs in the overlay network can exchange data traffic.

This article examines the structural components of routing and data policy in the Viptela overlay network.

Centralized and Localized Policy

The Viptela policy software design provides a clear separation between centralized and localized policy. In short, centralized policy is provisioned on the centralized vSmart controllers in the overlay network, and localized policy is provisioned on the vEdge routers, which sit at the network edge between a branch or enterprise site and a transport network, such as the Internet, MPLS, or metro Ethernet.

Centralized Policy

Centralized policy refers to policy provisioned on vSmart controllers, which are the centralized controllers in the Viptela overlay network. Centralized policy comprises two components:

- Control policy, which affects the overlay network–wide routing of traffic
- Data policy, which affects the data traffic flow throughout the VPN segments in the network

Centralized control policy applies to the network-wide routing of traffic by affecting the information that is stored in the vSmart controller's route table and that is advertised to the vEdge routers. The effects of centralized control policy are seen in how vEdge routers direct the overlay network's data traffic to its destination. The centralized control policy configuration itself remains on the vSmart controller and is never pushed to local routers.

Centralized data policy applies to the flow of data traffic throughout the VPNs in the overlay network. These policies can permit and restrict access based either on a 6-tuple match (source and destination IP addresses and ports, DSCP fields, and protocol) or on VPN membership. These policies are pushed to the affected vEdge routers.

Localized Policy

Localized policy refers to policy that is provisioned locally, on the vEdge routers in the overlay network

Localized control policy , which is also called route policy, affects the BGP and OSPF routing behavior on the site-local network.

Localized data policy allows you to provision access lists and apply them to a specific interface or interfaces on the router. Simple access lists permit and restrict access based on a 6-tuple match (source and destination IP addresses and ports, DSCP fields, and protocol), in the same way as with centralized data policy. Access lists also allow provisioning of class of service (CoS), policing, and mirroring, which control how data traffic flows out of and in to the router's interfaces and interface queues.

Control and Data Policy

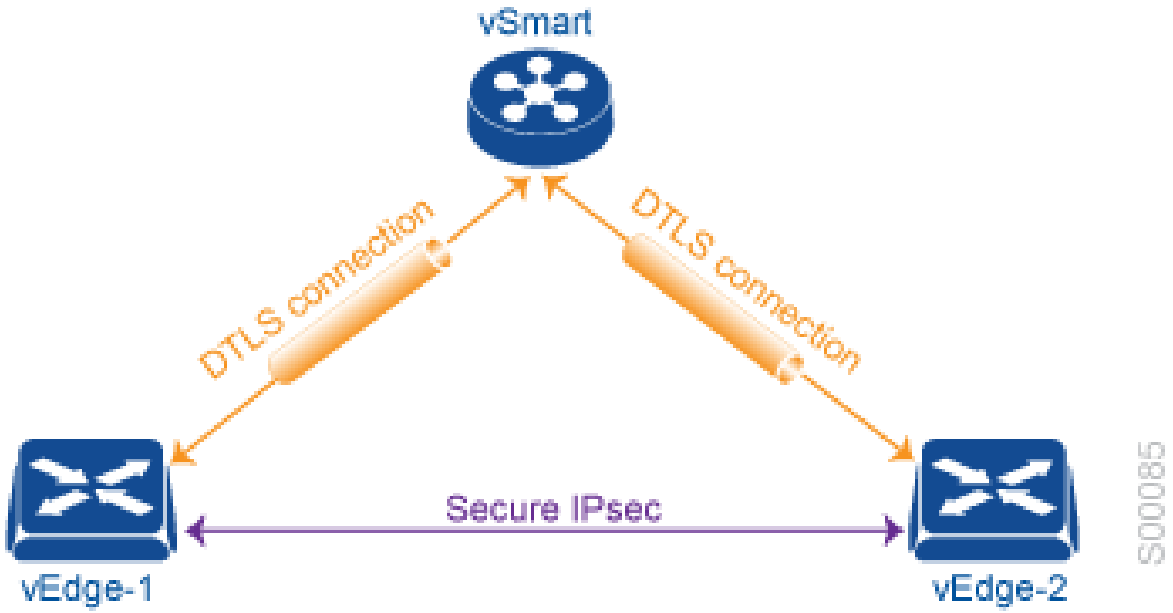
The Viptela network policy design provides a clean separation between control policy and data policy, to align with the network architecture in which the control and data planes are cleanly separated. Control policy is the equivalent of routing protocol policy, and data policy is equivalent to what are commonly called access control lists (ACLs) and firewall filters.

Control Policy

Control policy, which is similar to standard routing policy, operates on routes and routing information in the control plane of the overlay network. Centralized control policy, which is provisioned on the vSmart controller, is the Viptela technique for customizing network-wide

routing decisions that determine or influence routing paths through the overlay network. Local control policy, which is provisioned on a vEdge router, allows customization of routing decisions made by BGP and OSPF on site-local branch or enterprise networks.

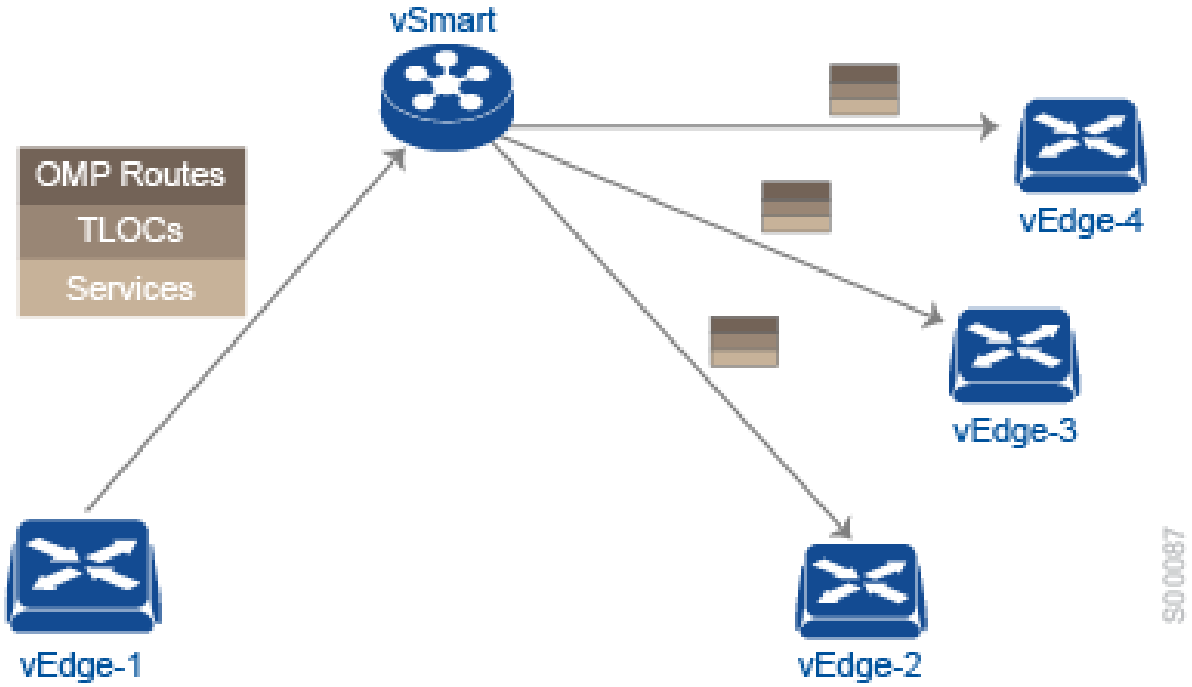
The routing information that forms the basis of centralized control policy is carried in Viptela route advertisements, which are transmitted on the DTLS or TLS control connections between vSmart controllers and vEdge routers. Centralized control policy determines which routes and route information are placed into the centralized route table on the vSmart controller and which routes and route information are advertised to the vEdge routers in the overlay network. Basic centralized control policy establish traffic engineering, to set the path that traffic takes through the network. Advanced control policy supports a number of features, including service chaining, which allows vEdge routers in the overlay network to share network services, such as firewalls and load balancers.



Centralized control policy affects the OMP routes that are distributed by the vSmart controller throughout the overlay network. The vSmart controller learns the overlay network topology from OMP routes that are advertised by the vEdge routers over the OMP sessions inside the DTLS or TLS connections between the vSmart controller and the routers. (The DTLS connections are shown in orange in the figure to the right).

Three types of OMP routes carry the information that the vSmart controller uses to determine the network topology:

- Viptela OMP routes, which are similar to IP route advertisements, advertise routing information that vEdge routers have learned from their local site and the local routing protocols (BGP and OSPF) to the vSmart controller. These routes are also referred to as OMP routes or vRoutes.
- TLOC routes carry overlay network–specific locator properties, including the IP address of the interface that connects to the transport network, a link color, which identifies a traffic flow, and the encapsulation type. (A TLOC, or transport location, is the physical location where a vEdge router connects to a transport network. It is identified primarily by IP address, link color, and encapsulation, but a number of other properties are associated with a TLOC.)
- Service routes advertise the network services, such as firewalls, available to VPN members at the vEdge router's local site.



By default, no centralized control policy is provisioned. In this bare, unpolicied network, all OMP routes are placed in the vSmart controller's route table as is, and the vSmart controller advertised all OMP routes, as is, to all vEdge routers in the same VPN in the network domain.

By provisioning centralized control policy, you can affect which OMP routes are placed in the vSmart controller's route table, what route information is advertised to the vEdge routers, and whether the OMP routes are modified before being put into the route table or before being advertised.

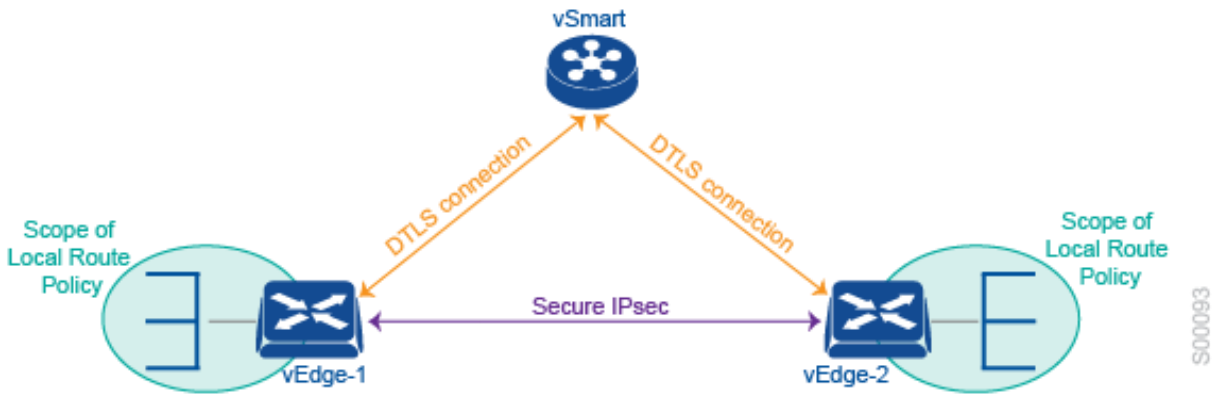
vEdge routers place all the route information learned from the vSmart controllers, as is, into their local route tables, for use when forwarding data traffic. Because the vSmart controller's role is to be the centralized routing system in the network, vEdge routers can never modify the OMP route information that they learn from the vSmart controllers.

The vSmart controller regularly receives OMP route advertisements from the vEdge routers and, after recalculating and updating the routing paths through the overlay network, it advertises new routing information to the vEdge routers.

The centralized control policy that you provision on the vSmart controller remains on the vSmart controller and is never downloaded to the vEdge routers. However, the routing decisions that result from centralized control policy are passed to the vEdge routers in the form of route advertisements, and so the affect of the control policy is reflected in how the vEdge routers direct data traffic to its destination.

A type of centralized control policy called **service chaining** allows data traffic to be routed through one or more network services, such as firewall, load balancer, and intrusion detection and prevention (IDP) devices, en route to its destination.

Localized control policy, which is provisioned locally on the vEdge routers, is called route policy. This policy is similar to the routing policies that you configure on a regular router, allowing you to modify the BGP and OSPF routing behavior on the site-local network. Whereas centralized control policy affects the routing behavior across the entire overlay network, route policy applies only to routing at the local branch.



Data Policy

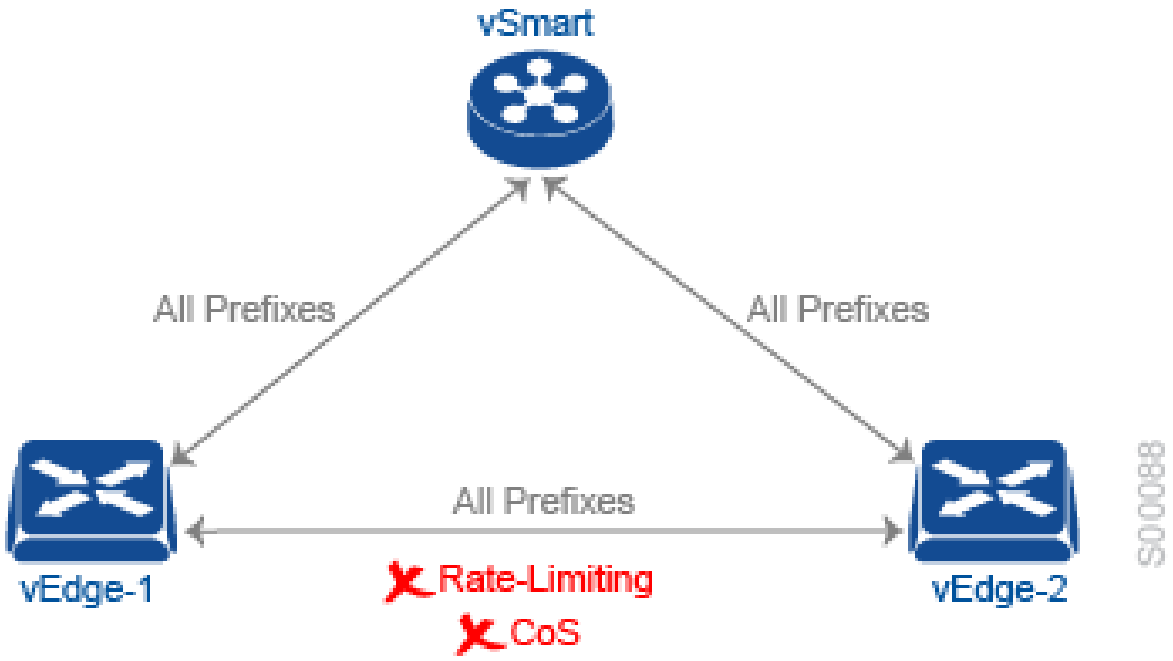
Data policy influences the flow of data traffic traversing the network based either on fields in the IP header of packets or the router interface on which the traffic is being transmitted or received. Data traffic travels over the IPsec connections between vEdge routers, shown in purple in the adjacent figure.

s00086.png

The Viptela architecture implements two types of data policy:

- Centralized data policy controls the flow of data traffic based on the source and destination addresses and ports and DSCP fields in the packet's IP header (referred to as a 5-tuple), and based on network segmentation and VPN membership. These types of data policy are provisioned centrally, on the vSmart controller, and they affect traffic flow across the entire network.
- Localized data policy controls the flow of data traffic into and out of interfaces and interface queues on a vEdge router. This type of data policy is provisioned locally, on the vEdge router, using access lists. It allows you to classify traffic and map different classes to different queues. It also allows you to mirror traffic and to police the rate at which data traffic is transmitted and received.

By default, no centralized data policy is provisioned. The result is that all prefixes within a VPN are reachable from anywhere in the VPN. Provisioning centralized data policy allows you to apply a 6-tuple filter that controls access between sources and destinations.



As with centralized control policy, you provision centralized data policy on the vSmart controller, and that configuration remains on the vSmart controller. The effects of data policy are reflected in how the vEdge routers direct data traffic to its destination. Unlike control policy, however, centralized data polices are pushed to the vEdge routers in a read-only fashion. They are not added to the router's configuration file, but you can view them from the CLI on the router.

With no access lists provisioned on a vEdge router, all data traffic is transmitted at line rate and with equal importance, using one of the interface's queues. Using access lists, you can provision class of service, which allows you to classify data traffic by importance, spread it across different interface queues, and control the rate at which different classes of traffic are transmitted. You can also provision packet mirroring and policing.

Basic and Advanced Policy

Finally, the design of the Viptela policy software distinguishes between basic and advanced policy. Basic policy allows you to influence or determine basic traffic flow through the overlay network. Here, you perform standard policy tasks, such as managing the paths along which traffic is routed through the network by influencing the OMP, BGP, and OSPF routes, and permitting or blocking traffic based on the address, port, and DSCP fields in the packet's IP header. You can also control the flow of data traffic into and out of a vEdge router's interfaces, enabling features such as class of service and queuing, mirroring, and policing.

Advanced features of Viptela policy software offer specialized policy-based network applications. Examples of these applications include the following:

- Service chaining, which allows data traffic to be redirected to shared firewall, intrusion detection and prevention (IDS), load balancer, and other devices before being delivered to its destination, rather than requiring a device at each branch site.
- Application-aware routing, which selects the best path for traffic based on real-time network and path performance characteristics.
- Cflowd, for monitoring traffic flow.
- Converting a vEdge router into a NAT device, to allow traffic destined for the Internet or other public network can exit directly from the vEdge router.

Additional Information

Centralized Control Policy
Centralized Data Policy
Localized Control Policy
Localized Data Policy

Viptela Policy Framework Basics

This article and the next, Policy Framework Applications , offer an orientation about the architecture of the Viptela policy software used to implement overlay network-wide policies. These policies are called *vSmart policy* or *centralized policy* , because you configure them centrally on a vSmart controller. vSmart policy affects the flow of both control plane traffic (routing updates carried by OMP and used by the vSmart controllers to determine the topology and status of the overlay network) and data plane traffic (data traffic that travels between the vEdge nodes across the overlay network).

With the Viptela software, you can also create vEdge routing policies on the vEdge edge routers. These policies are simply traditional routing policies that are associated with BGP or OSPF locally on the vEdge router. You use them in the traditional sense for controlling BGP and OSPF, for example, to affect the exchange of route information, to set route attributes, and to influence path selection. We do not discuss vEdge routing policy further these two orientation articles.

vSmart Policy Architecture Components

The vSmart policies that implement overlay network-wide policies are implemented on a vSmart controller. Because vSmart controllers are centralized devices, you can manage and maintain vSmart policies centrally, and you can ensure consistency in the enforcement of policy across the overlay network.

The implementation of vSmart policy is done by configuring the entire policy on the vSmart controller. vSmart policy configuration is accomplished with three building blocks:

- *Lists* define the targets of policy application or matching.
- *Policy definition* , or *policies* , controls aspects of control and forwarding. There are different types of policy, including:
  - app-route-policy (for application-aware routing)
  - cflowd-template (for cflowd flow monitoring)
  - control-policy (for routing and control plane information)
  - data-policy (for data traffic)
  - vpn-membership-policy (for limiting the scope of traffic to specific VPNs)
- *Policy application* controls what a policy is applied towards. Policy application is site-oriented, and is defined by a specific list called a site-list.

(In the bulleted list above and throughout this article, we occasionally use the name of the configuration command in text or figures when the meaning is clear. We do this to connect the discussion here with what you see in the software configuration. Configuration commands that are two or more English words use hyphens to separate the words. For example, app-route-policy is the command to configure an application-aware routing policy, and the control-policy command configures a control policy.)

You assemble these three building blocks to vSmart policy. More specifically, policy is the sum of one or more lists, one policy definition, and at least one policy applications, as shown in the figure below.

s00122.png

This figure lists the specific configuration keywords used to define lists and policies and to apply the policy. We discuss these in more detail later in this article.

vSmart Policy Operation

Knowing how vSmart policies are constructed, let's look at how they operate. Here, we'll look at the operation of three of the basic vSmart policies: control policy, data policy, and VPN membership policy. We'll example the operation of the other types of vSmart policy later in this article.

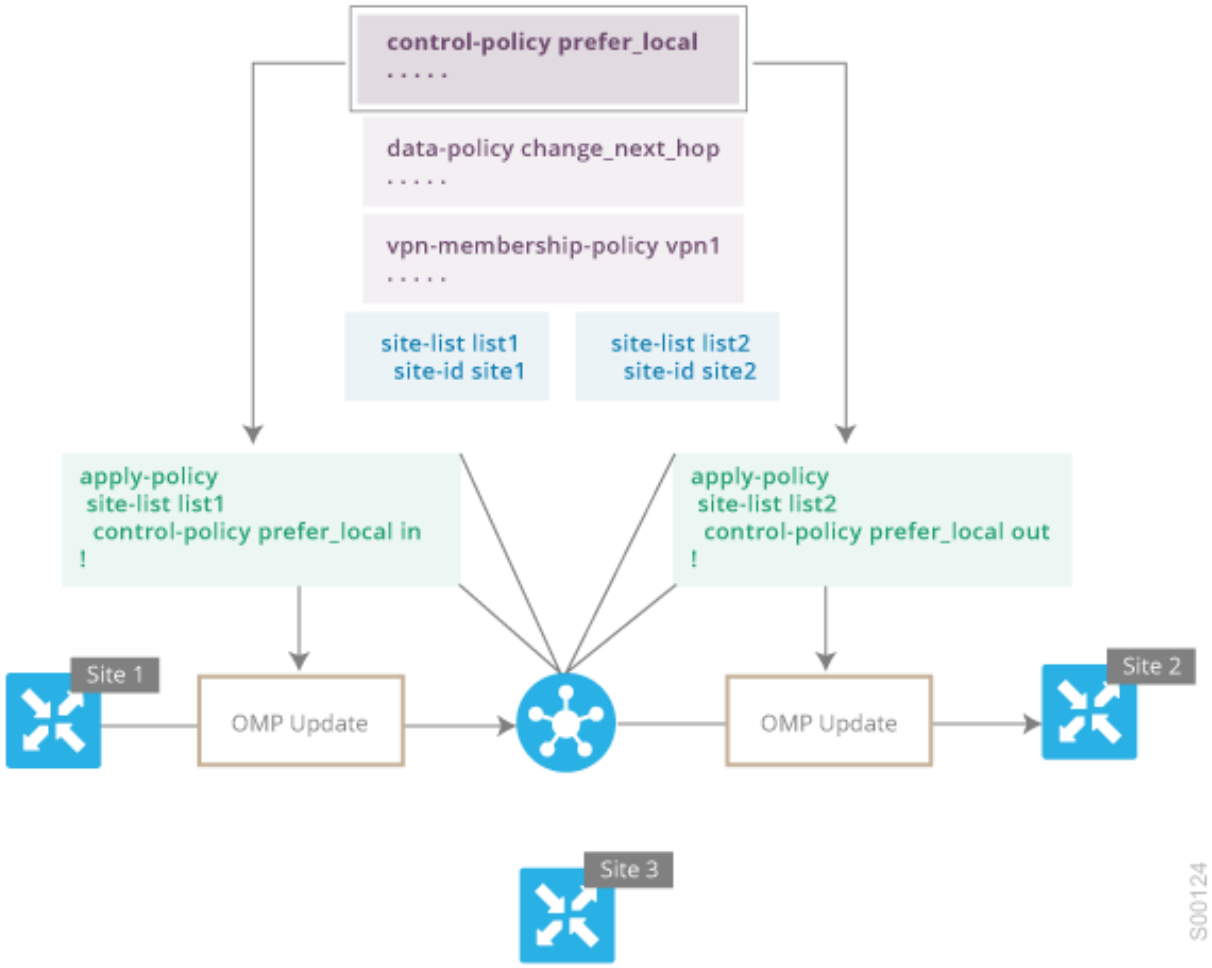
At a high level, control policy operates on routing information, which in the Viptela network is carried in OMP updates. Data policy affects data traffic, and VPN membership controls the distribution of VPN routing tables.

Control Policy Operation

Viptela devices periodically exchange OMP updates, which carry routing information pertaining to the overlay network. Two of the things that these updates contain are vRoute attributes and TLOC attributes. (The specific vRoute and TLOC attributes are discussed later in this article.) The vSmart controller uses these attributes from the OMP updates to determine the topology and status of the overlay network, and installs routing information about the overlay network into its route table. The controller then advertises the overlay topology to the vEdge routers in the network by sending OMP updates to them.

Control policy examines the vRoute and TLOC attributes carried in OMP updates and can modify attributes that match the policy. Any changes that results from control policy are applied directionally, either inbound or outbound (the directionality is from the point of view of the vSmart controller).

The figure below shows a control-policy named "prefer_local" that is configured on a vSmart controller and that is applied to Site 1 (via site-list list1) and to Site 2 (via site-list list2).



The upper left arrow shows the policy being applied to Site 1—more specifically, to **site-list list1** , which contains an entry for Site 1. The command to apply the policy is **control-policy prefer_local in** . The **in** keyword indicates an *inbound* policy: the policy is applied to OMP updates that are coming *in* to the vSmart controller from the vEdge router, which is inbound from the perspective of the controller. So, for all OMP updates that the Site 1 vEdge router sends to the vSmart controller, the "prefer_local" control policy is applied before the updates reach the route table on the vSmart controller. If any vRoute or TLOC attributes in an OMP update match the policy, any changes that result from the policy actions occur before the vSmart controller installs the OMP update information into its route table.

It is important to understand the effect of an inbound policy, because the route table on the vSmart controller is used to determine the topology of the overlay network. The vSmart controller then distributes this topology information, again via OMP updates, to all the vEdge routers in the network. Because applying policy in the inbound direction influences the information available to the vSmart controller to determine the network topology and network reachablity, modifying vRoute and TLOC attributes before they are placed in the controller's route table can provide broad influence over the flow of traffic throughout the overlay network.

On the right side of the figure above, we use the same "prefer_local" policy, but here apply it to Site 2 via the **control-policy prefer_local out** command. The **out** keyword in the command indicates an *outbound policy* , which means that the policy is applied to OMP updates that the vSmart controller is sending to the vEdge router at Site 2. Any changes that result from the policy occur *outbound* , after the information from the vSmart controller's route table has been placed into an OMP update and before the vEdge router receives the update. Again, note that the direction is outbound from the perpspective of the vSmart controller.

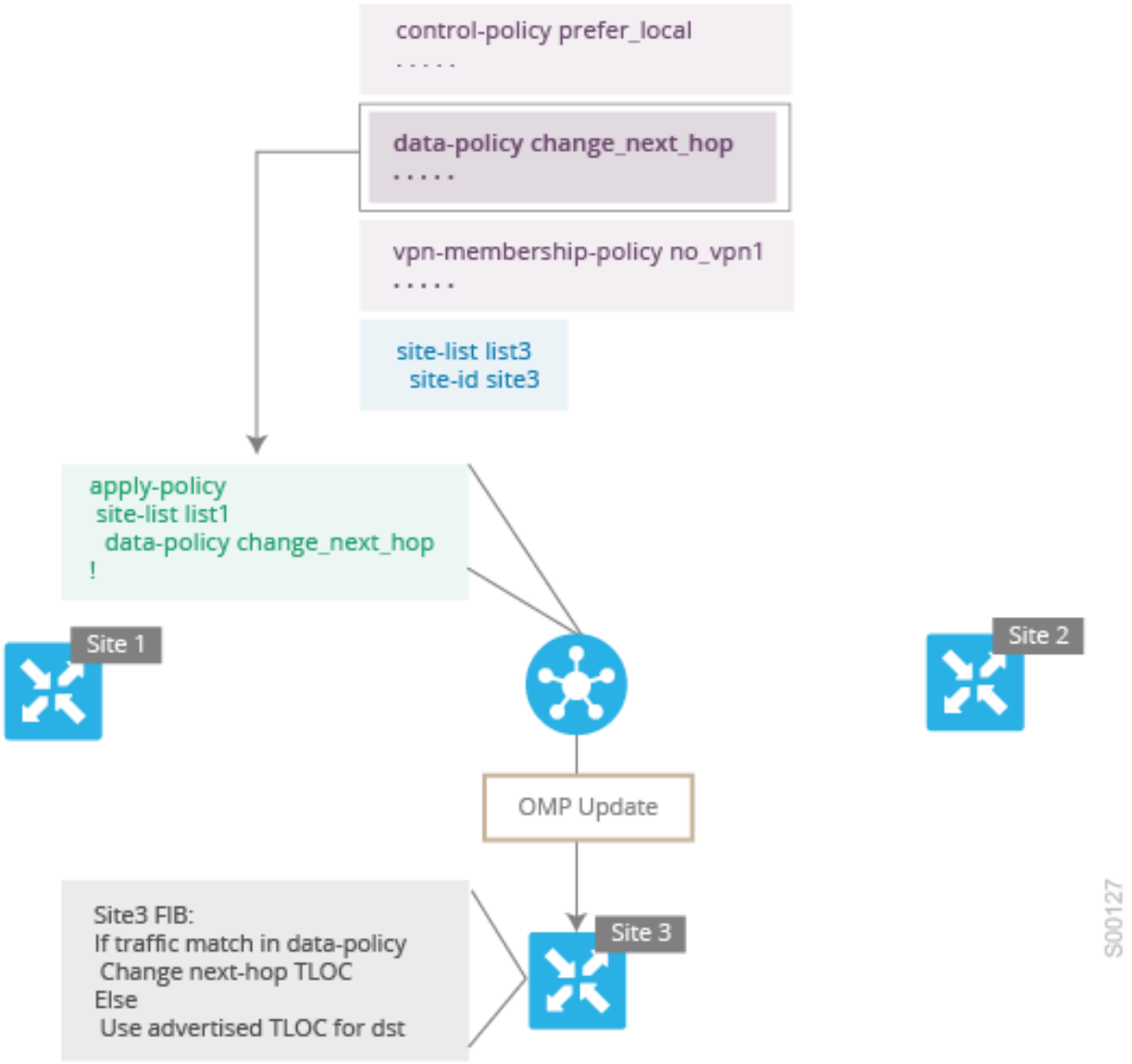
In contrast to an inbound policy, which affects the centralized route table on the vSmart controller and thus can have a broad effect on the route attributes advertised to all the vEdge routers in the overlay network, a control policy applied in the outbound direction influences only the route tables on the individual vEdge routers included in the site-list, so it generally has a more limited scope.

We point out again that in this figure, we are applying the same control policy (the "prefer_local" policy) to both the inbound and outbound OMP updates. However, the affects of applying the same policy inbound and outbound will be different. The usage shown in the figure illustrates the flexibility of the Viptela control policy design architecture and configuration.

Data Policy Operation

Data policy examines fields in the headers of data packets, looking at the source and destination addresses and ports, and the protocol and DSCP values, and for matching packets, it can modify the next hop in a variety of ways or apply a policer to the packets. Data policy is configured and applied on the vSmart controller, and then it is carried in OMP updates to the vEdge routers in the site-list that the policy is applied to. The match operation and any resultant actions are performed on the vEdge router as it transmits or receives data traffic.

In the figure below, a data policy named “change_next_hop” is applied to a list of sites that includes Site 3. The OMP update that the vSmart controller sends to the vEdge router at Site 3 includes this policy definition. When the vEdge router sends or receives data traffic that matches the policy, it changes the next hop to the specified TLOC. Nonmatching traffic is forwarded to the original next-hop TLOC.



In the apply-policy command for a data policy, you specify a direction from the perspective of the vEdge router. In the figure, the "all" direction applies the policy to data traffic transiting the tunnel interface, both what the vEdge router is sending and what it is receiving. You can limit the span of the policy to only incoming traffic (with a **data-policy change_next_hop from-tunnel** command) or to only outgoing traffic (with a **data-policy change_next_hop from-service** command).

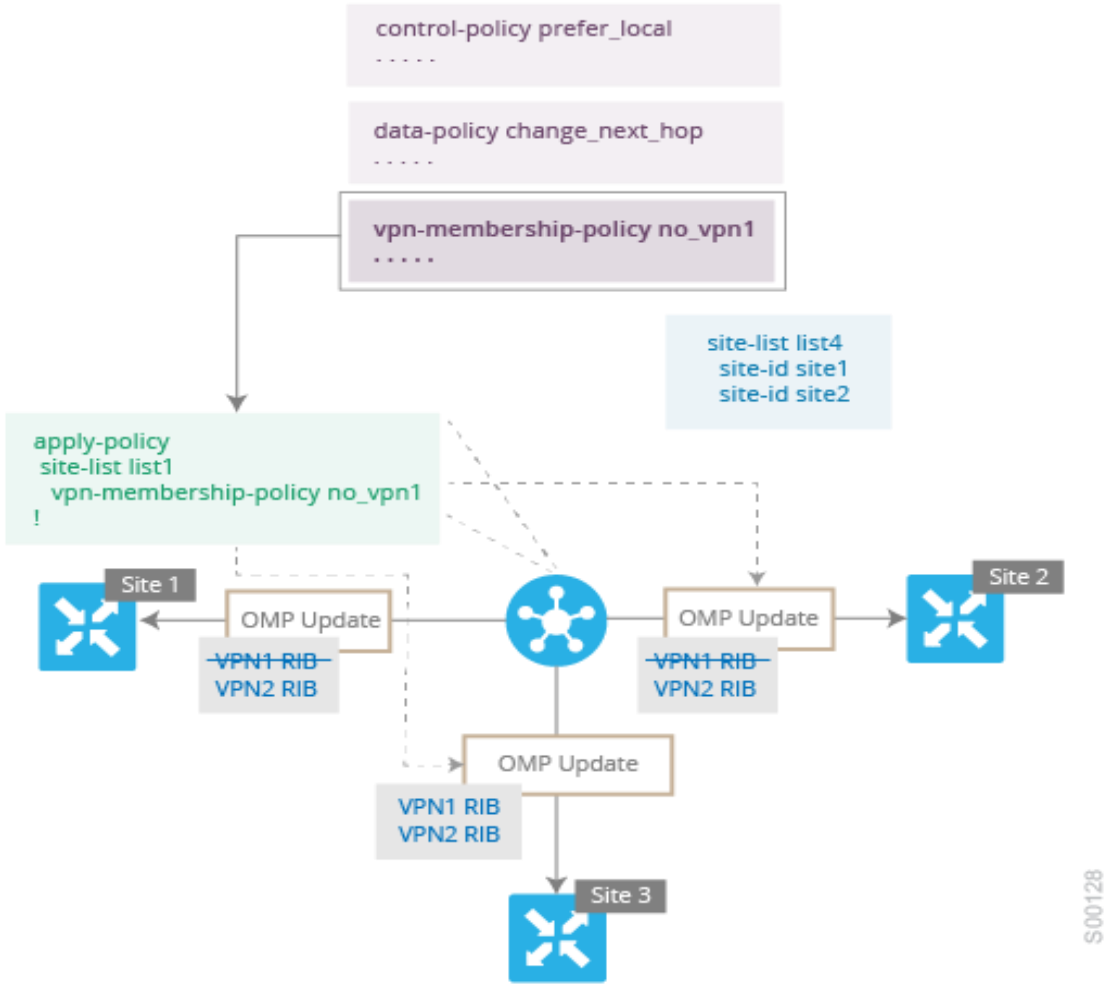
VPN Membership Policy Operation

VPN membership policy, as the name implies, affects the VPN route tables that are distributed to particular vEdge routers. In an overlay network with no VPN membership policy, the vSmart controller pushes the routes for all VPNs to all vEdge routers. If your business usage model restricts participation of specific vEdge routers in particular VPNs, a VPN membership policy is used to enforce this restriction.

The figure below illustrate how VPN membership policy works. This topology has three vEdge routers:

- The vEdge routers at Sites 1 and 2 service only VPN 2.
- The vEdge router at Site 3 services both VPN 1 and VPN 2.

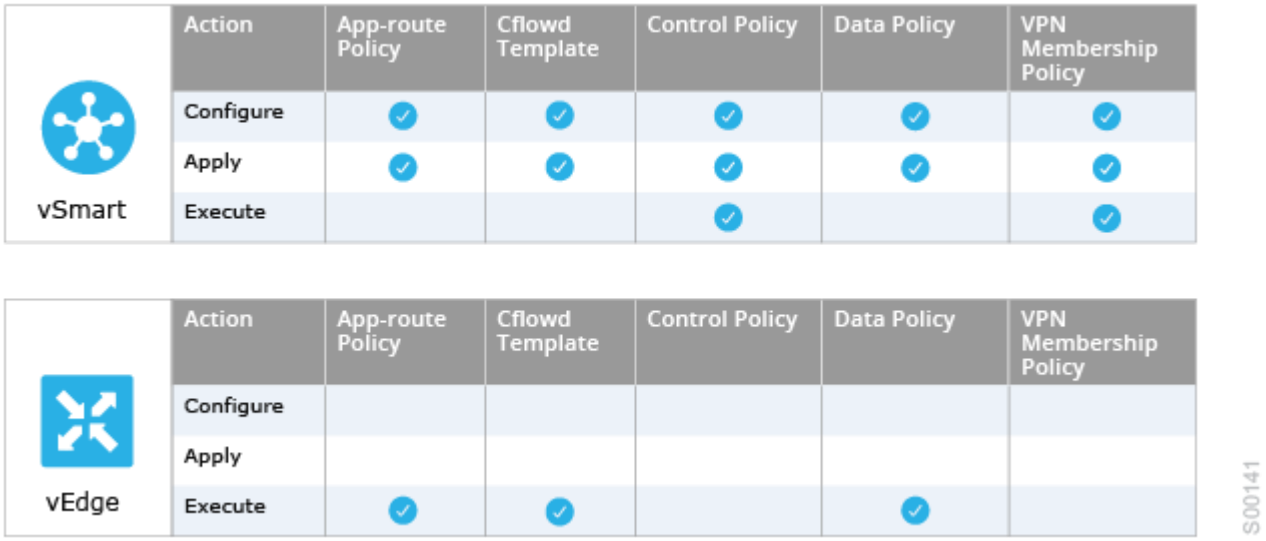
So here, we want the router at Site 3 to receive all route updates from the vSmart controller, because these updates are for both VPN 1 and VPN 2. However, because the other two routers service only VPN 2, we can filter the route updates sent to them, removing the routes associated with VPN 1 and sending only the ones that apply to VPN 2.



Notice that here, also, you don't set a direction when applying VPN membership policy. This vSmart controller always applies this type of policy to the OMP updates that it sends outwards to the vEdge routers.

Configuring and Executing vSmart Policies

All vSmart policies are configured on the vSmart controller, using a combination of policy definition and lists. All vSmart policies are also applied on the vSmart controller, with a combination of apply-policy and lists. However, where the actual vSmart policy executes depends on the type of policy, as shown in this figure:



For control policy and VPN membership policy, the entire policy configuration remains on the vSmart controller, and the actions taken as a result of routes or VPNs that match a policy are performed on the vSmart controller.

For the other three policy types—application-aware routing, cflowd templates, and data policy—the policies themselves are transmitted in OMP updates to the vEdge routers, and any actions taken as a result of the policies are performed on the vEdge routers.

vSmart Policy Components

Now let's discuss the software components used to construct policy in a bit more detail.

Lists

Lists are how you group related items so that you can reference them all together. Examples of items you put in lists are prefixes, TLOCs, VPNs, and overlay network sites. In vSmart policy, you invoke lists in two places: when you create a policy definition and when you apply a policy. Separating the definition of the related items from the definition of policy means that when you can add or remove items from a lists, you make the changes only in a single place: You do not have to make the changes through the policy definition. So if you add ten sites to your network and you want to apply an existing policy to them, you simply add the site identifiers to the site list. You can also change policy rules without having to manually modify the prefixes, VPNs, or other things that the rules apply to.

The illustration below shows the types of vSmart policy lists:

```
policy
 lists
  data-prefix-list app1
   ip-prefix 1.1.1.1/32 port 100
  !
  prefix-list pfx1
   ip-prefix 1.1.1.1/32
  !
  site-list site1
   site-id 100
  !
  tloc-list site1_tloc
   tloc 1.1.1.1 color mpls
  vpn-list vpn1
   vpn 1
  !
!
```

data-prefix-list is used in data-policy to define prefix and upper layer ports, either individually or jointly, for traffic matching.

prefix-list is used in control-policy to define prefixes for matching RIB entries.

site-list is used in control-policy to match source sites, and in apply-policy to define sites for policy application.

tloc-list is used in control-policy to define TLOCs for matching RIB entries and to apply redefined TLOCs to vRoutes.

vpn-list is used in control-policy to define prefixes for matching RIB entries, and in data-policy and app-route-policy to define VPNs for policy application.

Policy Definition

The policy definition is where you create the policy rules. You specify match conditions (route-related properties for control policy and data-related fields for data policy) and actions to perform when a match occurs. A policy contains match–action pairings that are numbered and that are examined in sequential order. When a match occurs, the action is performed, and the policy analysis on that route or packet terminates. Some types of policy definitions apply only to specific VPNs.

The following figure shows the components of the vSmart policy definition. These items are listed in the logical order you should use when designing policy, and this order is also how the items are displayed in the configuration, regardless of the order in which you add them to the configuration.

```
policy
 policy-type name
  vpn-list vpn-list
   sequence number
    match
     <route | tloc | vpn | other>
    !
    action <accept | reject |drop>
     set attribute value
    !
   default-action <reject | accept>
  !
 !
!
```

S001130

policy-type—which can be **control-policy** , **data-policy** , or **vpn-menbership** (as well as a few other kewords that we discuss later)—dictates the type of policy. Each type has a particular syntax and a particular set of match conditions and settable actions.

vpn-list is used by data-policy and app-route-policy to list the VPNs for which the policy is applicable.

sequence defines each sequential step of the policy by sequence number.

match decides what entity to match on in the specific policy sequence.

action determines the action that corresponds to the preceding match statement.

default-action is the action to take for any entity that is not matched in any sequence of the policy. By default, the action is set to reject.

Policy Application

For a policy definition to take effect, you associate it with sites in the overlay network.

```
apply-policy
 site-list name
  control-policy name <in|out>
 !
 site-list name
  data-policy name
  vpn-membership name
 !
!
```

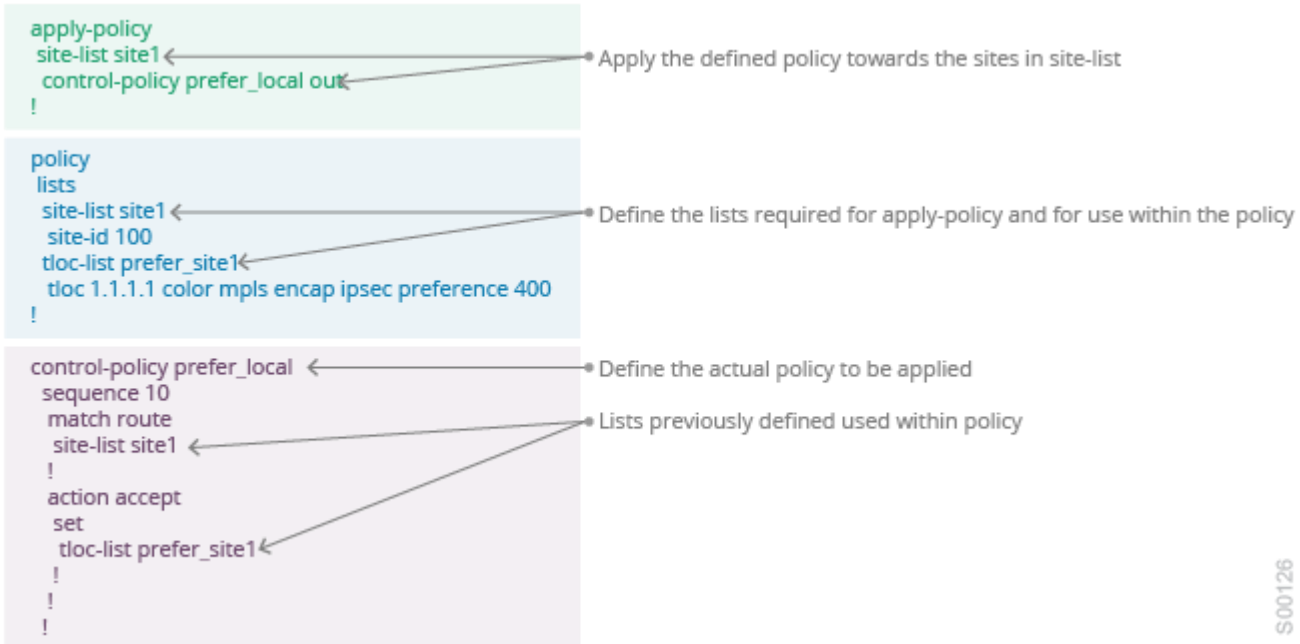
The following are the configuration components:

site-list determines the sites to which a given policy is applies. The direction (in | out) applies only to control-policy.

The policy type— **control-policy** , **data-policy** , **vpn-membership** —and name refer to an already configured policy to be applied to the sites specified in the site-list for the section.

Policy Example

Now, let's put together a complete policy, which consists of lists, policy definition, and policy application. The example illustrated below creates two lists (a site-list and a tloc-list), defines one policy (a control policy), and applies the policy to the site-list. In the figure, the items are listed as they are presented in the node configuration. In a normal configuration process, you create lists first (group together all the things you want to use), then define the policy itself (define what things you want to do), and finally apply the policy (specify the sites that the configured policy affects).



TLOC Attributes Used in Policies

A transport location, or TLOC, defines a specific interface in the overlay network. Each TLOC consists of a set of attributes that are exchanged in OMP updates among the Viptela devices. Each TLOC is uniquely identified by a 3-tuple of IP address, color, and encapsulation. Other attributes can be associated with a TLOC.

The TLOC attributes listed below can be matched or set in vSmart policies.

TLOC Attribute

Function

Application Point

Set By

Modify By

Address (IP address)

system-ip address of the source device on which the interface is located.

Configuration on source device

control-policy
data-policy

Carrier

Identifier of the carrier type. It primarily indicates whether the transport is public or private.

Configuration on source device

control-policy

Color

Identifier of the TLOC type.

Configuration on source device

control-policy
data-policy

Domain ID

Identifier of the overlay network domain.

Configuration on source device

control-policy

Encapsulation

Tunnel encapsulation, either IPsec or GRE.

Configuration on source device

control-policy
data-policy

Originator

system-ip address of originating node.

Configuration on any originator

control-policy

Preference

OMP path-selection preference. A higher value is a more preferred path.

Configuration on source device

control-policy

Site ID

Identification for a give site. A site can have multiple nodes or TLOCs.

Configuration on source device

control-policy

Tag

Identifier of TLOC on any arbitrary basis.

Configuration on source device

control-policy

vRoute Attributes Used in Policies

A Viptela route, or vRoute, defines a route in the overlay network. A vRoute, which is similar to a standard IP route, has a number attributes such as TLOC and VPN. Viptela devices exchange vRoutes in OMP updates.

The vRoutes attributes listed below can be matched or set in vSmart policies.

vRoute Attribute

Function

Application Point

Set By

Modify By

Origin

Source of the route, either BGP, OSPF, connected, static.

Source device

control-policy

Originator

Source of the update carrying the route.

Any originator

control-policy

Preference

OMP path-selection preference. A higher value is a more preferred path.

Configuration on source device or policy

control-policy

Service

Advertised service associated with the vRoute.

Configuration on source device

control-policy

Site ID

Identifier for a give site. A site can have multiple nodes or TLOCs.

Configuration on source device

control-policy

Tag

Identification on any arbitrary basis.

Configuration on source device

control-policy

TLOC

TLOC used as next hop for the vRoute.

Configuration on source device or policy

control-policy
data-policy

VPN

VPN to which the vRoute belongs.

Configuration on source device or policy

control-policy
data-policy

Understanding vSmart Policy Processing and Application

Understanding how vSmart policy is processed and applied allows fo proper design of policy and evaluation of how policy is being implemented across the overlay network.

Policy is processed in this way:

- A policy definition consists of a numbered, ordered sequence of match–action pairings. Within each policy, the pairings are processed in sequential order, starting with the lowest number and incrementing.

- As soon as a match occurs, the matched entity is subject to the configured action of the sequence and is then no longer subject to continued processing.
- Any entity not matched in a sequence is subject to the default action for the policy. By default, this action is reject.

vSmart policy is applied on a per-site-list basis, so:

- When applying policy to a site-list, you can apply only one of each type of policy. For example, you can have one control-policy and one data-policy, or one control-policy in and one control-policy out. You cannot have two data policies or two outbound control policies.
- Because a site-list is a grouping of many sites, you should be careful about including a site in more than one site-list, and in general, we recommend that you not do this at all. You should take special care when a site-list includes a range of site identifiers, to ensure that there is no overlap. If the same site is part of two site-lists and the same type of policy is applied to both site-lists, the policy behavior will be unpredictable and possibly catastrophic.

Control-policy is unidirectional, being applied either inbound to the vSmart controller or outbound from it. When control-policy is needed in both directions, configure two control policies. Data-policy is directional and can be applied either to traffic received from the service side of the vEdge router, traffic received from tunnel side, or both. VPN membership policy is always applied to traffic outbound from the vSmart controller.

Control-policy remains on the vSmart controller and affects routes that the controller sends and receives.

Data-policy is sent to vEdge routers in the site-lite. The policy is sent in OMP updates, and it affects the data traffic that the routers send and receive.

When any node in the overlay network makes a routing decision, it uses any and all available routing information. In the overlay network, it is the vSmart controller that distributes routing information to the vEdge nodes. In a network deployment that has two or more vSmart controller, each controller acts independently to disseminate routing information to other vSmart controllers and to vEdge routers in the overlay network. So, to ensure that vSmart policy has the desired effect in the overlay network, each vSmart controller must be configured with the same policy, and the policy must be applied identically. What this means is that for any given policy, you must configure the identical policy and apply it identically across all the vSmart controllers.

Additional Information

Configuring Centralized Control Policy
Configuring Centralized Data Policy
Policy Framework Applications

Policy Framework Applications

This article builds on the discussion in the previous article, Viptela Policy Framework Basics , describing policy applications that control the distribution of route information in the overlay network, and that direct data traffic flows through the network. All the policies we discuss here are vSmart policies, also called centralized policies, which are policies that are configured on the vSmart controller.

We discuss the following applications of the vSmart policy framework:

- Application-aware routing policy looks at the network and path characteristics of the data plane connections between vEdge routers, compares them to configured SLA parameters, and computes optimal paths for data traffic.
- Control policy uses OMP vRoute and TLOC information to enable services and engineer specialized routing applications.
- Data policy
- VPN membership policy

Application-Aware Routing Policies

Application-aware routing tracks packet loss and latency on the data plane connections between vEdge routers, to compute optimal paths for data traffic in the overlay network. The loss and latency data supplements the standard routing parameters—such as route prefixes, metrics, and link-state information—in determining traffic paths through the network.

Application-aware routing policy is defined by means of a specially designed vSmart policy, called an app-route-policy. This policy specifies the performance characteristics in the form an SLA, which defines the required latency and loss conditions on the data plane connection that trigger the policy to take effect.

Application-aware routing is created in three portions of the configuration:

- Configure the sla-class, which defines the required latency and loss for the traffic that is to be affected by a given app-route-policy.
- Configure the app-route-policy, which specifies the traffic that is to belong to an sla-class. (This is done in a fashion similar to a data-policy.) The app-route-policy references a vpn-list to dictate which VPNs at the listed sites benefit from the policy.
- Apply the app-route-policy towards the desired overlay network sites.

The following is an example of an app-route-policy configuration:

```
apply-policy
 site-list site1
  app-route-policy app_route
 !
!
```

```
policy
 sla-class EF
   loss 2
   latency 100
!
app-route-policy app_route
 vpn-list app_vpn
  sequence 10
   match dscp 46
!
action
  sla-class EF
  !
 !
!
```

```
lists
  site-list site 100
    site-id 100
!
vpn-list app_vpn
  vpn 1
 !
!
```

S00151

sla-class defines the required performance metrics for the application.

Here, the sla-class is named EF. For the performance metrics, the maximum acceptable packet loss is 2 percent and the maximum acceptable packet latency is 100 milliseconds. If either one of these values is greater than the configured value, the data tunnel is considered invalid. In other words, if the latency is greater than 100 milliseconds or if the packet loss is greater than 2 percent, that tunnel is not considered as a valid data path.

vpn-list defines the VPNs where the policy is applied to at the target sites. You create the vpn-list in the lists section, and you apply it in the definition of the app-route-policy itself.

Here, the app-route-policy applies only to the vpn-list called app_vpn, which affects only data traffic in VPN 1.

app-route-policy defines the actual application route policy. In the policy definition, you link the matched traffic (in the match portion of the policy) with the required sla-class, which is called in the action portion of the policy.

Here, the action portion of the app-route-policy calls the previously defined sla-class EF.

Cflowd Flow Data Collection

Cflowd monitors traffic flowing through vEdge routers in the overlay network and exports flow information to a collector, where it can be processed by an IPFIX analyzer. Cflowd periodically sends template reports to flow collector. These reports contain information about the flow and data extracted from the IP headers of transiting flows.

Cflowd flow collection is enabled by means of a vSmart policy, specifically, by a vSmart data policy. The parameters for capturing and exporting flow data are defined in two sections of the policy:

- A cflowd-template configures the flow cache behavior and flow export. For the flow cache, the cflowd-template defines how often the sampled flows should be sent to a collection. For the flow export, the cflowd-template specifies the location of the flow collection. You must configure a cflowd-template, but it need not contain any parameters. With no parameters, the data flow cache on vEdge nodes is managed using default settings, and no flow export occurs.
- A data-policy selects the traffic subject to flow data collection. The data-policy can be configured to be very specific or as a general flow collection filter, depending on requirements.

Both the cflowd-template and the data-policy components are controlled and configured on the vSmart controller, and they are both distributed from the vSmart controller to the affected vEdge routers. This design eases the configuration and application of traffic flow monitoring.

The following is an example of configuring a cflowd-template and a data-policy:

```
apply-policy
 site-list site100
  data-policy cflowd_data all
  cflowd-template cflowd_temp
 !
!
```

```
policy
 data-policy cflowd_data
  vpn-list cflowd_vpn
   sequence 10
    match
     protocol 17
    !
    action accept
     cflowd
    !
   !
   default-action drop
  !
 !
```

```
cflowd-template cflowd_temp
 flow-active-timeout   60
 flow-inactive-timeout 60
 collector vpn 100 address 1.1.1.1 port 4739 transport transport_udp
!
!
```

S00132

data-policy (in the purple policy section of the configuration) defines the actual traffic subject to flow data collection. Here, data-policy cflowd_data matches all UDP traffic (protocol 17) and applies flow monitoring to it. The default-action is drop, so all non-UDP data traffic is dropped.

data-policy (in the green apply-policy section) applies data-policy cflowd_data to the vEdge routers at sites that are part of the site-list named site100. The **all** option applies the policy to data traffic entering and leaving the vEdge routers.

cflowd-template (in the brown policy section) creates a template to manage cache management and flow export settings. Here, we define the template called cflowd_temp that gives the location flow collector location (in VPN 100, at address 1.1.1.1 and port 4739) and exports flow information every 60 seconds for both active and inactive flows.

cflowd-template (in the green apply-policy section) applies the cflowd template to the vEdge routers in site-list site100. The cflowd-template and data-policy are applied at the same time.

Control Policy Applications and Services

vSmart control policies are put in place to influence routing in the overlay domain. Using OMP routing information, these policies can enable services and engineer specialized routing applications. Control policy is a powerful tool for any type of path management. Having it be centralized and managed on vSmart controllers greatly simplifies policy operations throughout the network.

In this section, we describe the following types of vSmart control policies:

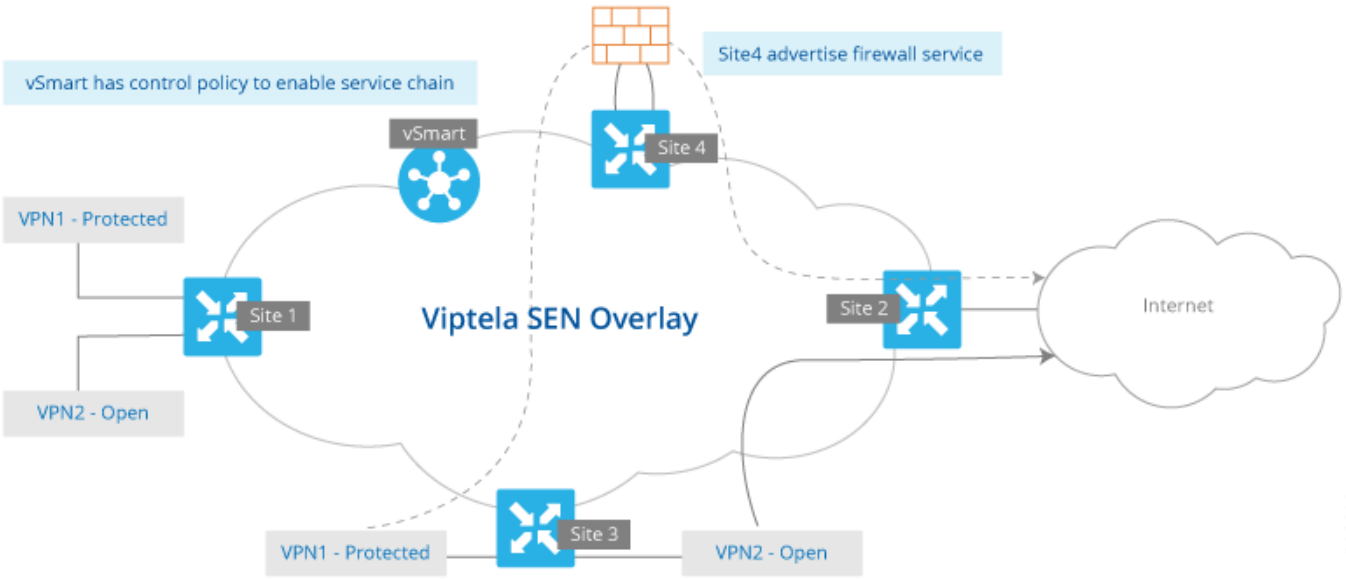
- Service chaining
- Traffic engineering
- Extranet VPNs
- Service and path affinity
- Arbitrary VPN topologies

Service Chaining with Control Policy

When security devices and resources are located at one site in the network and routers and hosts dispersed throughout the network need to access these resources, you can use vSmart policies to redirect data traffic to those resources before it is forwarded towards its final destination. In the Viptela policy framework, this process is called *service chaining* , because an intermediate destination at which a service is located is being attached, or *chained* , to a route. These centralized security devices and resources are called *services* .

vSmart policy that implement service chaining selectively directs traffic to standard services—firewalls, Intrusion Detection Systems (IDSs), and Identity Providers (IDPs)—or to custom services that you define.

As an example of how service chaining can be used, traffic from some departments (in the figure below, those in VPN 1) may require firewall protection when traveling to and from the Internet or external networks, while traffic from other departments may not (in VPN 2). Departments in VPN 1 are located at Sites 1 and 3, and their traffic needs to use the firewall service at Site 4. The Viptela solution applies a vSmart service-chaining policy to the affected VPN.



The configuration for the service chaining has the following components:

- The service originator advertises the firewall service.
- A control-policy directs traffic in VPN 1 that is headed from the overlay network to the Internet through the firewall service. This is the policy that is applied to Sites 1 and 3.
- A control-policy directs incoming traffic destined for VPN 1 through the firewall service. Control-policy is unidirectional, which is why we need to define two control policies. This policy is applied to Site 2.
- Application of the two policies.

First, the service originator advertises the firewall service. The originator is the vEdge router at Site 4, which advertises a firewall device at IP address 1.1.1.1 that is available to VPN 1. This vEdge router advertises this information in the OMP service routes that it sends to the vSmart controller.

```
vpn 1
 service FW address 1.1.1.1
```

S00148

The vSmart control policies have these components:

```
policy
 lists
  vpn-list closed_vpn
   vpn 1
  !
  site-list branch_sites
   site-id site1, site3
  !
  site-list site2
   site-id site2
 !
!
```

```
control-policy service-chain
  sequence 10
   match route
    origin bgp-external
    vpn-list closed_vpn
   !
   action accept
    set
     service FW
    !
   !
  !
  default-action accept
 !
!
```

```
control-policy service-chain-return
  sequence 10
   match route
    site-list branch_sites
   !
   action accept
    set
     service FW
    !
   !
  !
  default-action accept
 !
!
```

```
apply-policy
 site-list branch_sites
  control-policy service-chain out
 !
 site-list site2
  control-policy service-chain-return out
 !
```

S00150

In **vpn-list** , the list called closed_vpn contains VPN 1, the protected VPN. We need this list for the match condition in control-policy service-chain. If we decide later that traffic from other VPNs needs firewall protection, we can simply add these VPNs to the closed_vpn VPN list.

In **site-list** , we create the lists used in the **apply-policy** commands. The site-list branch_sites holds the two sites in VPN 1, and site-list site2 contains the site that connects to the Internet.

control-policy service-chain defines the target routes for which the firewall service is needed. These routes are external BGP prefixes in VPN 1. A default-action of accept means that nonmatching prefixes, such as those in other VPNs, pass through the policy unchanged.

control-policy service-chain-return defines the target routes for the firewall service in the return path. These are the prefixes in Sites 1 and 3 that are in VPN 1. Again, the default-action accept means that nonmatching prefixes pass through the policy unchanged.

apply-policy site-list branch_sites applies the control-policy service-chain for the upstream firewall service to Sites 1 and 3. This control-policy has the following effect on outbound traffic towards the Internet:

- For traffic originating from VPN 1 at Sites 1 and 3, the next hop for outgoing BGP prefixes is changed to point to the location where the firewall service is hosted.
- The firewall receives the traffic from its directly connected vEdge router, processes it, and then follows its route table, which points back to the connected vEdge router. It is very likely that the firewall device simply has a default route and nothing more in its route table.
- Because no policy is applied to the vEdge router adjacent to the firewall device, when it performs a route table lookup, the next hop for the BGP prefixes is the node that originally sourced the traffic into OMP. This node is the vEdge router at Site 2, the router that originally learned the prefixes from the Internet and advertised them into OMP.

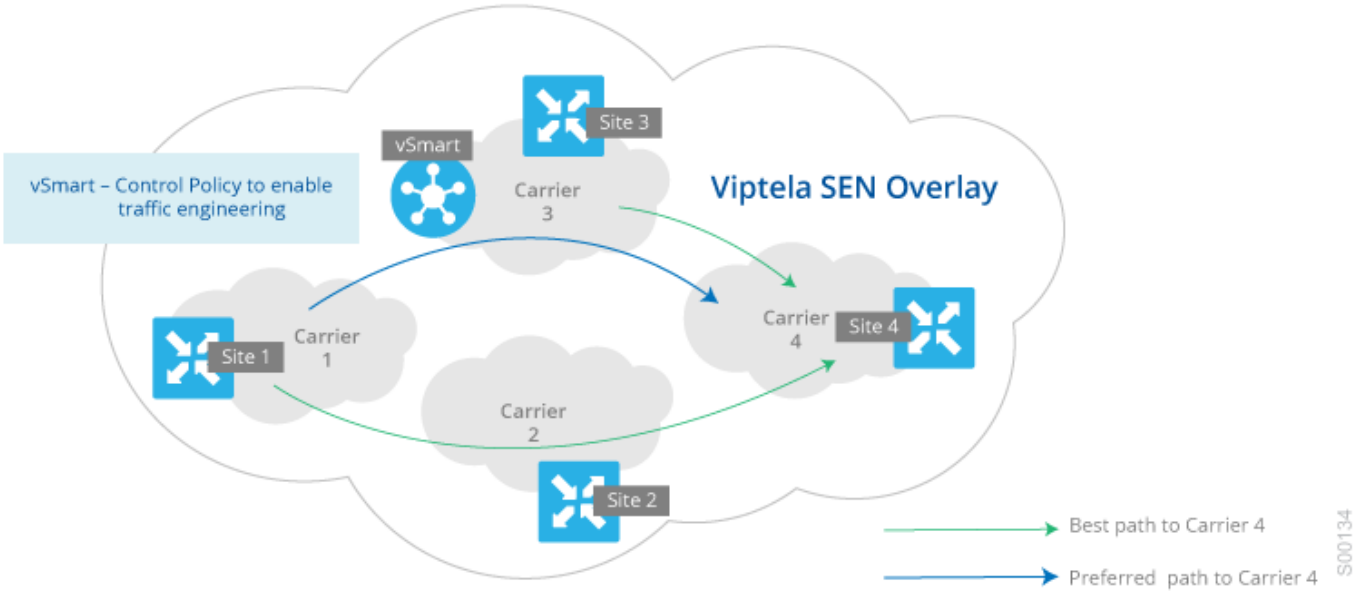
To summarize, in the upstream direction (from the overlay network to the Internet), the original next hop for BGP prefixes is used for all traffic except for VPN 1 traffic from Sites 1 and 3.

apply-policy site-list site2 applies the control-policy service-chain-return for incoming (downstream) firewall service. This control-policy has the following effect on traffic inbound from the Internet:

- For prefixes that were originally sourced from Sites 1 and 3 in VPN 1, Site 2 changes the next hop to point to the firewall service.
- The firewall receives the traffic from the directly connected vEdge router, processes it, and, per its route table, sends it back to its connected vEdge router.
- Because no policy has been applied to the vEdge router at Site 4, the original next hops for the OMP-sourced prefixes are used, and the traffic is directed towards Sites 1 and 3.

Traffic Engineering

Traffic engineering is a mechanism for controlling traffic flow through links in the network, optimizing the path to suit the needs of the network. MPLS is one way to integrate traffic engineering into Layer 3 applications, such as routing IP traffic. In the Viptela network, you can use vSmart policy directly to traffic-engineer paths through the overlay network. In the example illustrated below, even though the best path from Carrier 1 to Carrier 4 is through Carrier 2, we instead want traffic to go via Carrier 3.



The vSmart policy configuration to establish traffic engineering has these components:

```
policy
 lists
  site-list site1
   site-id site1
  !
  site-list site4
   site-id site4
 !
 tloc-list prefer_carrier3
  tloc site3 color public-internet preference 400
  tloc site4 color public-internet preference 200
!
```

```
 control-policy traffic_eng
  sequence 10
   match route
    site-list site4
   !
   action accept
    set
     tloc-list prefer_carrier3
    !
   !
  !
  default-action accept
 !
!
```

```
apply-policy
 site-list site1
  control-policy traffic_eng out
 !
!
```

S00135

site-list has two lists. One identifies the sites in the overlay network where we apply the policy (Site 1). The second is used in the match condition, to identify traffic headed towards Site 4.

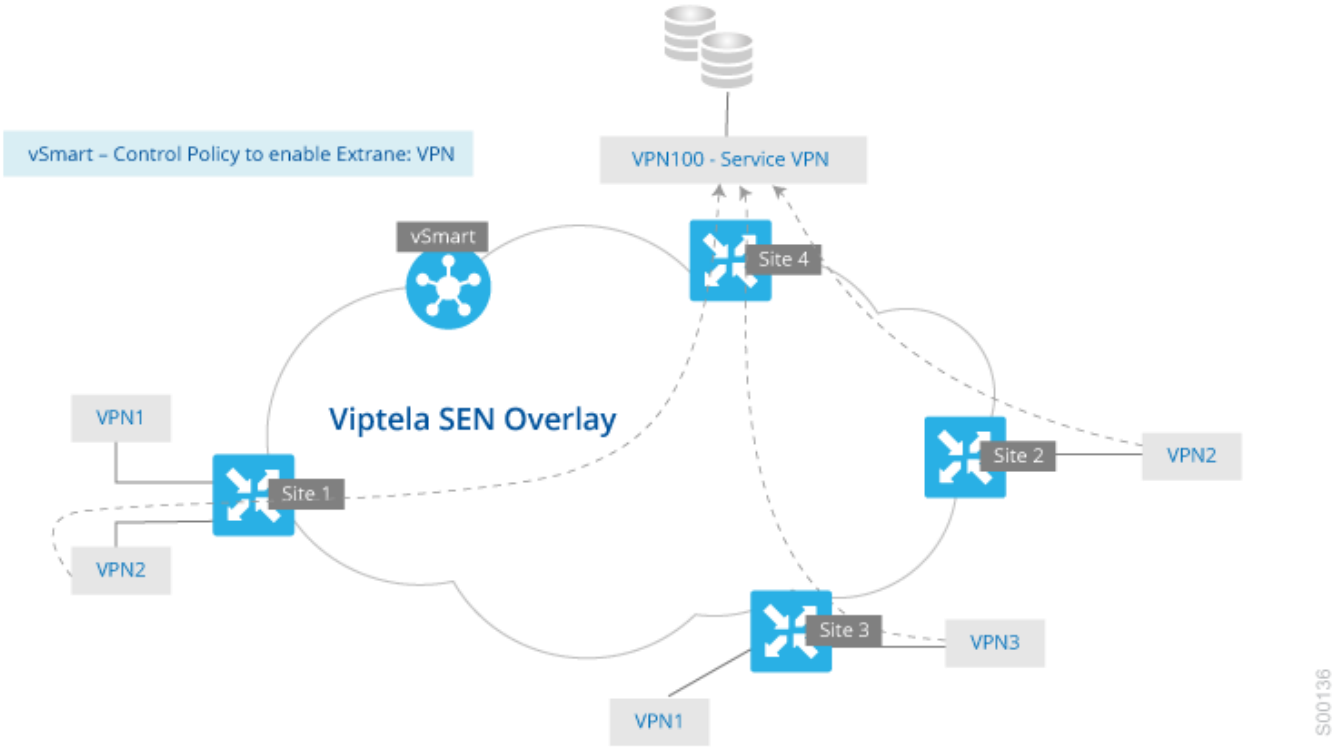
tloc-list is where you actually engineer the traffic path. You do this by setting the preference on two of the TLOCs. (In the configuration commands, you identify the TLOCs by their IP addresses, not by "site3" and "site4".) The preference to use TLOC 3, and so to have traffic go via Carrier 3, is higher that the preference to use TLOC 4 (and go via Carrier 2 to Carrier 4).

control-policy traffic_eng defines the target prefixes for which traffic engineering is required. The policy matches prefixes associated with routes to Site 4. The action in this policy is to change the data packet's TLOC properties to those defined in the tloc-list prefer_carrier3 list, so the traffic prefers Site 3 (and Carrier 3) as its next hop.

apply-policy applies the traffic engineering policy towards the target sites in site-list site1. The policy is applied in the outbound direction, which means that any route changes that occur as a result of the policy are done after the vSmart controller sends the OMP route update and before the update is received by the vEdge router at Site 1.

Extranet VPNs

When services that reside in a VPN must be shared across users residing in multiple other VPNs (see illustration below), you can create a vSmart extranet VPN control policy. This policy advertises the prefixes in the service VPNs to the other (client) VPNs, and it advertises the client prefixes to the service VPN, so that the vEdge routers connected to these VPNs have reachability information for each other.



In this example, the shared services, which are file servers are located in VPN 100, and users in VPNs 2 and 3 need access to the file servers (and users in VPN 1 do not).

We need three lists—two VPN lists and one site list. One VPN lists contains the member of the service VPN (VPN 100), and the second the members of the client VPNs (VPNs 2 and 3). Both VPN lists are used in the control-policy match conditions. The third list, which contains all the sites in the overlay network, is used in the apply-policy statement.

```
policy
  lists
    vpn-list client_vpn
      vpn 2,3
    !
    vpn-list service_vpn
      vpn 100
    !
    site-list all_sites
```

```
      site-id site1
      site-id site2
      site-id site3
      site-id site4
    !
  !
!
```

The control-policy has two sequence entries, one that matches the client_vpn routes and exports (advertises) them to the service VPN, and one that matches the service_vpn routes and advertises them to the client VPNs:

```
policy
  control-policy extranet
    sequence 10
      match route
        vpn-list client_vpn
      !
      action accept
        export-to
          vpn-list service_vpn
        !
      !
    !
    sequence 20
      match route
        vpn-list service_vpn
      !
      action accept
        export-to
          vpn-list client_vpn
        !
      !
    !
  !
  default-action accept
  !
!
```

To have the vSmart control-policy take effect, apply it inbound direction for prefixes for each affect VPN to be imported into the vSmart controller's route table:

```
apply-policy
  site-list all_sites
    control-policy extranet in
  !
!
```

Data Policy

vSmart data policy is a powerful tool for any type of data plane–centered traffic management. Because it is centralized and managed on a vSmart controller data policy greatly simplifies policy operations. Data policies are used to enable a number of services, including:

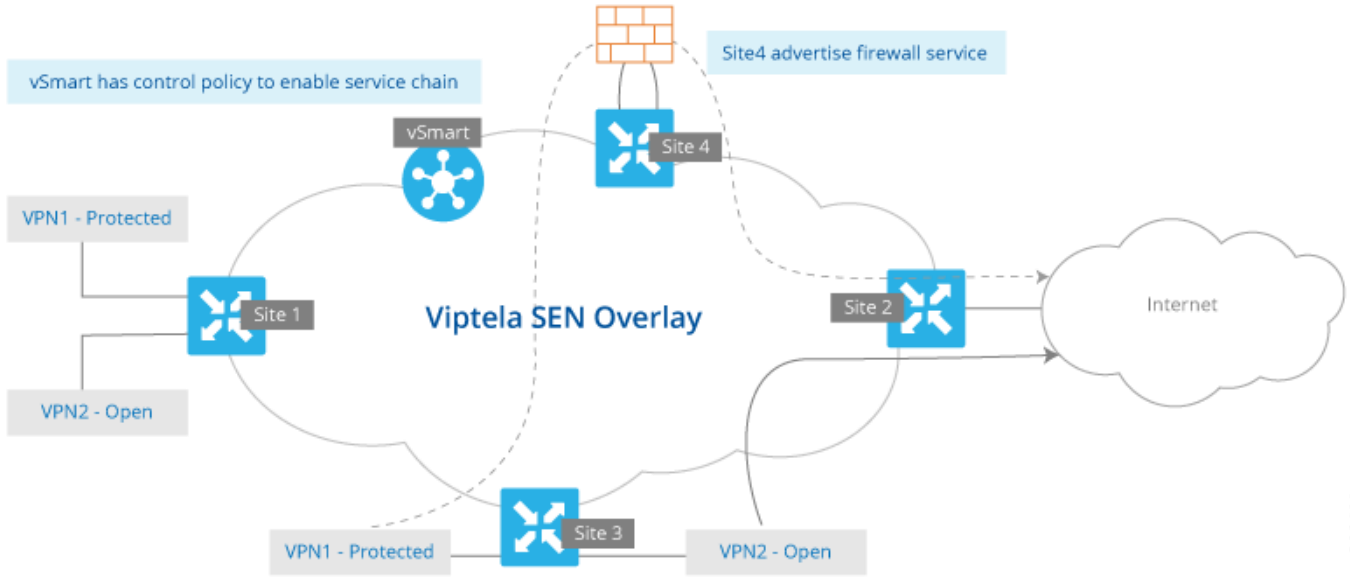
- Service chaining
- Cflowd traffic monitoring
- Traffic policing and counting

You configure and apply data policies on the vSmart controller, which then pushes the policy to the vEdge routers. It is the routers that enforce the configured policy in the data plane. A data policy acts on an entire VPN and is not interface-specific.

Some of the policy applications that can be enabled with control policies can also be enabled with data policies (and also with traditional routing policy). Enabling applications with data policies instead of control policies can be simpler to administer, because the policies apply only to a single node and they do not interact across the overlay network.

Service Chaining with Data Policy

Earlier, we showed a way to configure service chaining with control-policy. For the same network, here is the configuration done with data-policy.



As before, the service originator (Site 4) advertises the firewall service. Here is the configuration for the vEdge router at Site 4:

```
vpn 1
 service FW address 1.1.1.1
```

S00148

Then we configure the actual policy on the vSmart controller: '

```
policy
 lists
  vpn-list closed_vpn
   vpn 1
  !
  site-list branch_sites
  site-id site1
  site-id site3
  !
 !
!
```

```
policy
 data-policy service-chain
 vpn-list closed_vpn
  sequence 10
   match
    dscp    10
   !
   action accept
    set
     service FW
    !
   !
  !
  default-action accept
 !
!
```

```
 apply-policy
  site-list branch_sites
   data-policy service-chain all
  !
 !
```

S00147

First we create two lists:

site-list identifies the branch sites. These are the sites included in the apply-policy command, the sites that receive the data-policy. Sites 1 and 3 in our figure serve the protected VPN, VPN 1, so the list contains these two sites.

vpn-list , as in the control-policy version, identifies the VPNs whose traffic needs to pass through the firewall service.

data-policy service-chain defines the target DSCP value for traffic that requires the firewall service.

Here, the policy, called closed_vpn, sets the criteria that packets in the protected VPN must match to be directed to the firewall service. The match in this example in done on QoS, using the DSCP value of 10.

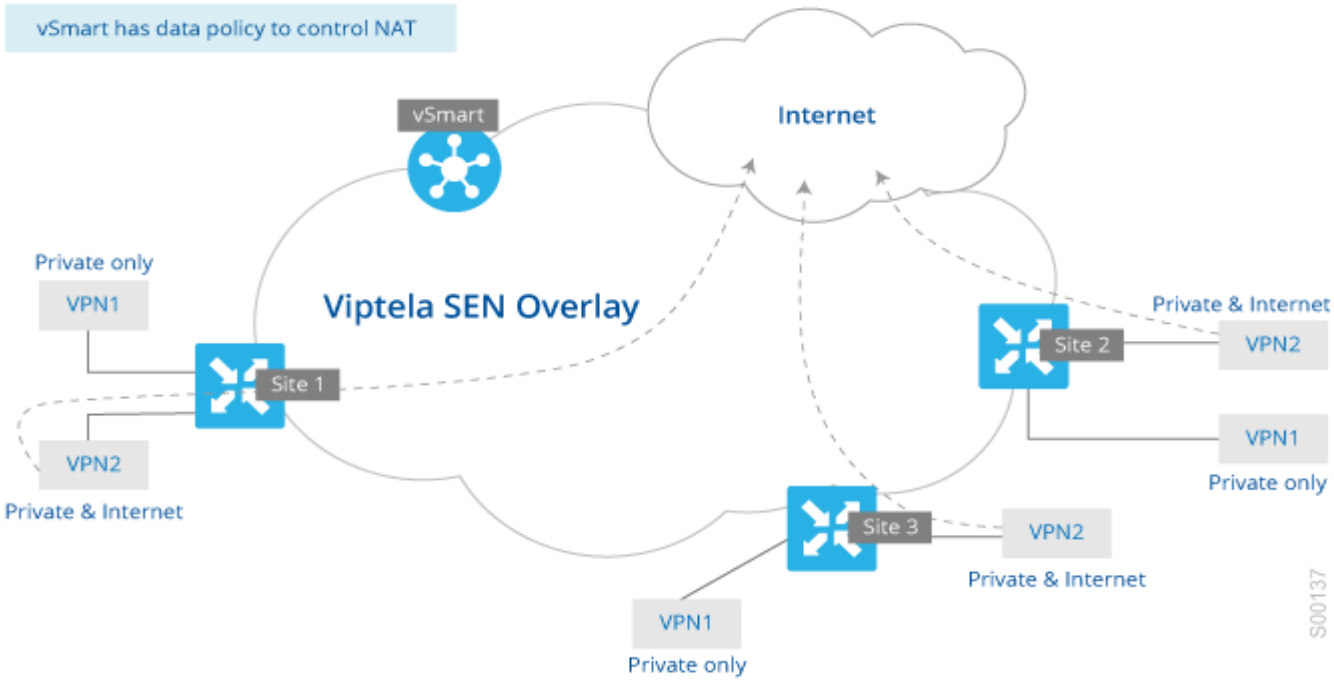
apply-policy site-list branch_sites applies the data policy to the branch sites in the list. data-policy is always applied on a directional& basis: from-service, from-tunnel, or both (**all**).

In a situation where a control-policy was considered to be a better fit for a service-chaining application, it could be deployed in conjunction with a data-policy to enable cflowd data flow collection.

Using a NAT for Local Internet Exit

Rather than have a single exit point from the overlay network to the Internet, vSmart data-policy can provide local Internet exit from vEdge routers. You implement this using a data-policy that includes a NAT directive. The data-policy is configured on the vSmart controller, so local Internet exit is managed centrally.

The figure below illustrates a topology that provides local Internet exit to departments that are part of VPN 2. The users in these departments are located at all three of the sites in the overlay network.



```
policy
 lists
  data-prefix-list internet_bound
   ip-prefix 1.1.1.0/24
   ip-prefix 2.2.2.0/24
  !
  vpn-list Internet_vpn
   vpn 2
  !
  site-list branch_sites
   site-id site1
   site-id site2
   site-id site3
  !
 !
!
```

```
policy
 data-policy internet_nat
  vpn-list Internet_vpn
   sequence 10
    match
     source-data-prefix-list internet_bound
    !
    action accept
     nat use-vpn 0
    !
   !
   default-action accept
  !
 !
```

```
apply-policy
 site-list branch_sites
  data-policy internet_nat from-service
 !
!
```

S00145

Here are the configuration components that set up the NAT function on the vEdge routers so that traffic passing destined for the Internet can travel directly from the vEdge router to the Internet, without having to go to a centralized or other site before exiting to the Internet:

data-prefix-list identifies the source IP prefixes whose traffic is destined for the NAT, and hence for the Internet

vpn-list identifies the affected VPNs.

site-list groups together the three sites that participate in VPN 1.

data-policy in the **policy** section directs matching traffic towards the NAT, and hence towards the Internet.

data-policy in the **apply-policy** section is applied in the from-service direction to match incoming traffic from the service side to the vEdge router.

VPN Membership Policy

The default behavior of the VIptela OMP architecture is to advertise any configured VPN to any node on which it is configured. This design automatically establishes connectivity without unnecessary configuration and operational overhead. However, certain VPNs may be of a sensitive nature such that their membership must be tightly controlled.

VPN membership policy serves to restrict the distribution of VPN information from the vSmart controllers to those VPNs that are explicitly approved. With a VPN membership policy, a vEdge router that is not explicitly allowed to participate in a VPN may have the VPN configured, but it can see only local connectivity and routing information.

You can use VPN membership policy to establish both allowed list and block list behavior.

```
policy
 lists
  site-list sites_1
   site-id site1
   site-id site2
  !
  site-list sites_2
   site-id site3
   site-id site4
  !
  vpn-list sites_1
   vpn 10, 20
  !
  vpn-list sites_2
   vpn 30, 40
  !
 !
!
```

```
policy
 vpn-membership acme_1
  sequence 10
   match vpn-list sites_1
   action accept
   !
  !
  default-action reject
 !
 vpn-membership acme_2
  sequence 10
   match vpn-list sites_2
   action accept
   !
  !
  default-action reject
 !
!
```

```
apply-policy
 site-list sites_1
  vpn-membership acme_1
 !
 site-list sites_2
  vpn-membership acme_2
 !
!
```

S00143

For VPN membership policies, you configure the following components:

vpn-lists define the VPN match data. The example shown to the left has two VPN lists: **vpn-list sites_1** contains VPNs 10 and 20, and **vpn-list sites_2** contains VPNs 30 and 40.

site-lists are used in two places: in the policy math condition and when applying the policy to the affected sites in the overlay network.

vpn-membership defines the policies themselves, which can act as either allowed lists or block lists for VPN filtering.

apply-policy applies the VPN membership policies in both directions, to determine which VPNs are allowed from a given site.

Additional Information

Configuring Centralized Control Policy
Configuring Centralized Data Policy
Viptela Policy Framework Basics

Centralized Control Policy

Centralized control policy is policy that is configured on a vSmart controller (hence, it is centralized and operates on the control plane traffic in the Viptela overlay network, influencing the determination of routing paths through the overlay network. Centralized control policy affects the route information that is exchanged between vSmart controllers and vEdge routers in the Viptela overlay network.

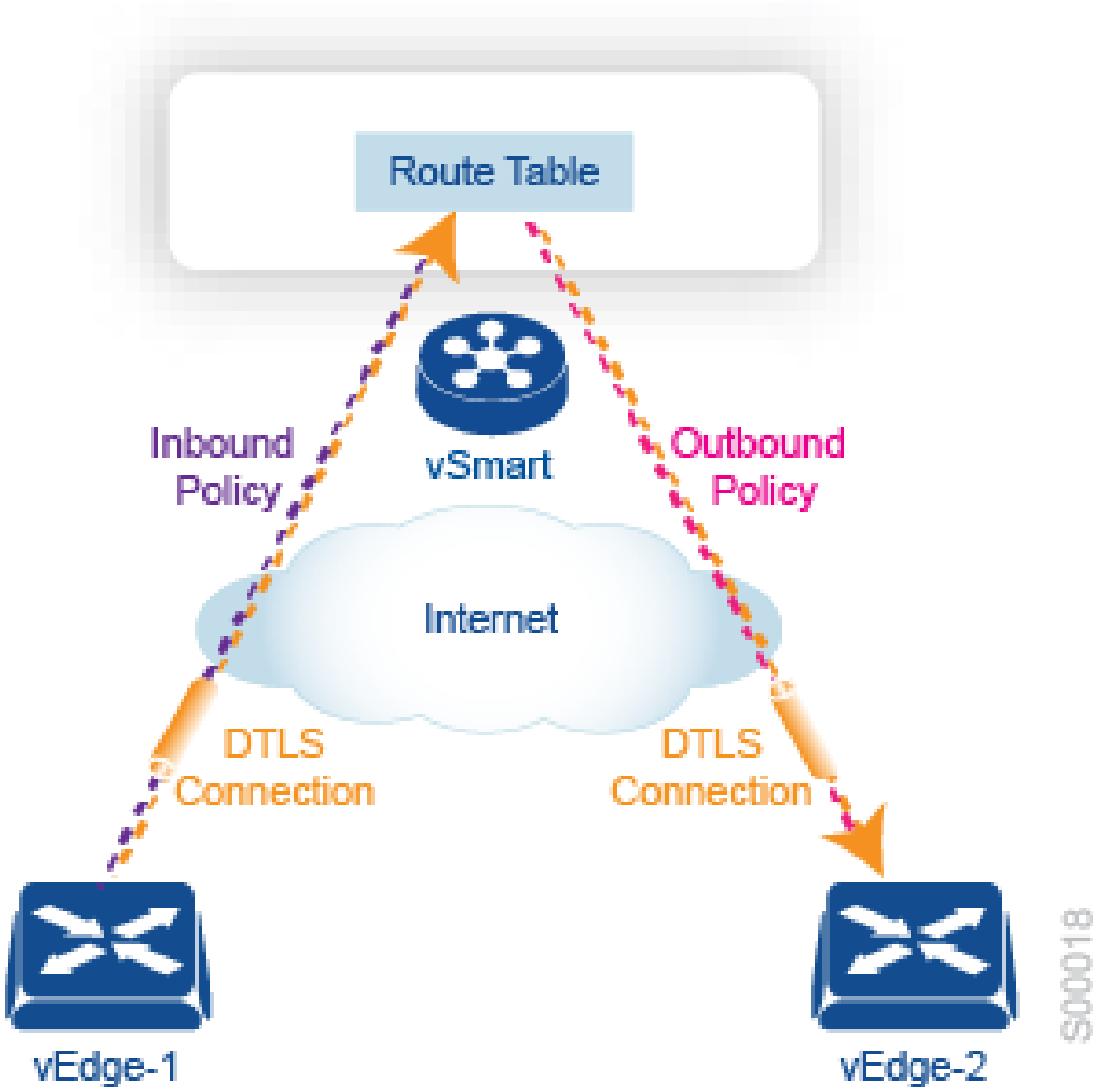
Centralized Control Policy Overview

In the Viptela network architecture, centralized control policy is handled by the vSmart controller, which effectively is the routing engine of the Viptela network. The vSmart controller is the centralized manager of network-wide routes, maintaining a primary route table for these routes. The vSmart controller builds its route table based on the route information advertised by the vEdge routers in its domain, using these routes to discover the network topology and to determine the best paths to network destinations. The vSmart controller distributes route information from its route table to the vEdge routers in its domain, and the vEdge routers use these routes to forward data traffic through the network. The result of this architecture is that networking-wide routing decisions and routing policy are orchestrated by a central authority instead of being implemented hop by hop, by the devices in the network.

Centralized control policy allows you to influence the network routes advertised by the vSmart controllers. This type of policy, which is provisioned centrally on the vSmart controller, affects both the route information that the vSmart controller stores in its primary route table and the route information that it distributes to the vEdge routers.

Centralized control policy is provisioned and applied only on the vSmart controller. The control policy configuration itself is never pushed to vEdge routers in the overlay network. What is pushed to the vEdge routers, via the Overlay Management Protocol (OMP), are the results of the control policy, which the vEdge routers then install in their local route tables and use for forwarding data traffic. This design means that the distribution of network-wide routes is always administered centrally, using policies designed by network administrators. These policies are always implemented by centralized vSmart controllers, which are responsible for orchestrating the routing decisions in the Viptela overlay network.

Within a network domain, the network topology map on all vSmart controllers must be synchronized. To support this, you must configure identical policies on all the vSmart controllers in the domain.



All centralized control plane traffic, including route information, is carried by OMP peering sessions that run within the secure, permanent DTLS connections between vEdge routers and the vSmart controllers in their domain. The end points of an OMP peering session are identified by the system IDs of the Viptela devices, and the peering sessions carry the site ID, which identifies the site in which the device is located. A DTLS connection and the OMP session running over it remain active as long as the two peers are operational.

Control policy can be applied both inbound, to the route advertisements that the vSmart controllers receives from vEdge routers, and outbound, to advertisements that it sends to them. Inbound policy controls which routes and route information are installed in the local routing database on the vSmart controller, and whether this information is installed as-is or is modified. Outbound control policy is applied after a route is retrieved from the routing database, but before vSmart controller advertises it, and affects whether the route information is advertised as-is or is modified.

Route Types

The vSmart controller learns the network topology from OMP routes, which are Viptela-specific routes carried by OMP. There are three types of OMP routes:

- Viptela OMP routes—These routes carry prefix information that the vEdge router learns from the routing protocols running on its local network, including routes learned from BGP and OSPF, as well direct, connected, and static routes. OMP advertises OMP routes to the vSmart controller by means of an OMP route SAFI (Subsequent Address Family Identifier). These routes are commonly simply called OMP routes.
- TLOC routes—These routes carry properties associated with transport locations, which are the physical points at which the vEdge routers connect to the WAN or the transport network. Properties that identify a TLOC include the IP address of the WAN interface and a color that identifies a particular traffic flow. OMP advertises TLOC routes using a TLOC SAFI.
- Service routes—These routes identify network services, such as firewalls and IDPs, that are available on the local-site network to which the vEdge router is connection. OMP advertises these routes using a service SAFI.

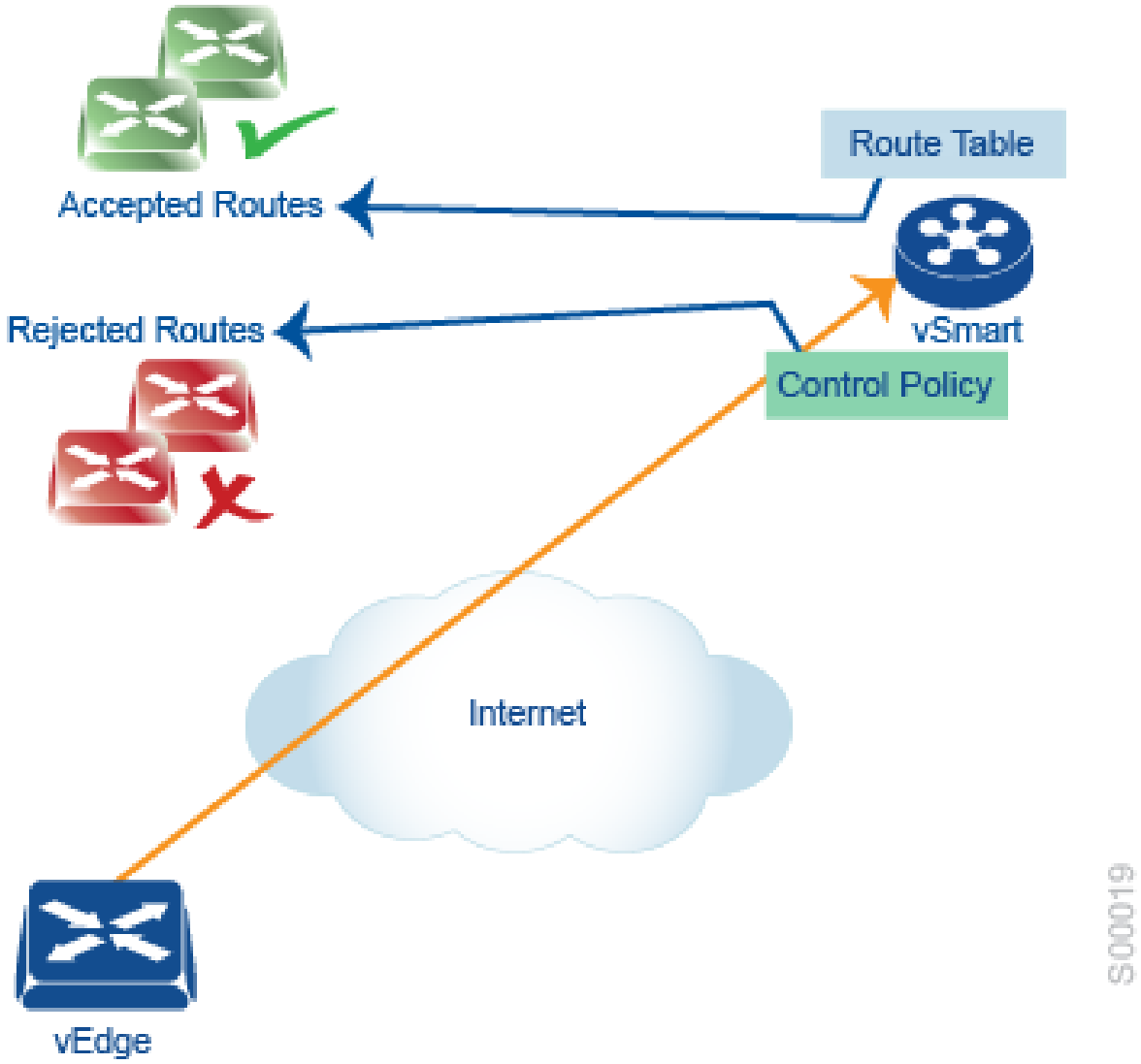
A straightforward way to see the difference in these three types of routes is by using the various **show omp** operational commands when you are logged in to the CLI on a vSmart controller or a vEdge router. The **show omp routes** command displays information sorted by prefix, the **show omp services** command display route information sorted by service, and the **show omp tlocs** command sorts route information by TLOC.

Default Behavior without Centralized Control Policy

By default, no centralized control policy is provisioned on the vSmart controller. This results in the following route advertisement and redistribution behavior within a domain:

- All vEdge routers redistribute all the route-related prefixes that they learn from their site-local network to the vSmart controller. This route information is carried by OMP route advertisements that are sent over the DTLS connection between the vEdge router and the vSmart controller. If a domain contains multiple vSmart controllers, the vEdge routers send all OMP route advertisements to all the controllers.
- All vEdge routers send send all TLOC routes to the vSmart controller or controllers in their domain, using OMP.
- All vEdge routers send all service routes to advertise any network services, such as firewalls and IDPs, that are available at the local site where the vEdge router is located. Again, these are carried by OMP.
- The vSmart controller accepts, as is, all the OMP, TLOC, and service routes that it receives from all the vEdge routers in its domain, storing the information in its route table. The vSmart controller tracks which OMP routes, TLOCs, and services belong to which VPNs. The vSmart controller uses all the routes to develop a topology map of the network and to determine routing paths for data traffic through the overlay network.
- The vSmart controller redistributes all information learned from the OMP, TLOC, and service routes in a particular VPN to all vEdge routers in the same VPN.
- The vEdge routers regularly send route updates to the vSmart controller.
- The vSmart controller recalculates routing paths, updates its route table, and advertises new and changed routing information to all the vEdge routers.

Behavior Changes with Centralized Control Policy



When you do not want to redistribute all route information to all vEdge outers in a domain, or when you want to modify the route information that is stored in the vSmart controller's route table or that is advertised by the vSmart controller, you design and provision centralized control policy. To activate the control policy, you apply it to specific sites in the overlay network in either the inbound or the outbound direction. The direction is with respect to the vSmart controller. All provisioning of centralized control policy is done on the vSmart controller.

Applying a centralized control policy in the inbound direction filters or modifies the routes being advertised by the vEdge router before they are placed in the route table on the vSmart controller. As the first step in the process, routes are either accepted or rejected. Accepted routes are installed in the route table on the vSmart controller either as received or as modified by the control policy. Routes that are rejected by a control policy are silently discarded.

Applying a control policy in outbound direction filters or modifies the routes that the vSmart controller redistributes to the vEdge routers. As the first step of an outbound policy, routes are either accepted or rejected. For accepted routes, centralized control policy can modify the routes before they are distributed by the vSmart controller. Routes that are rejected by an outbound policy are not advertised.

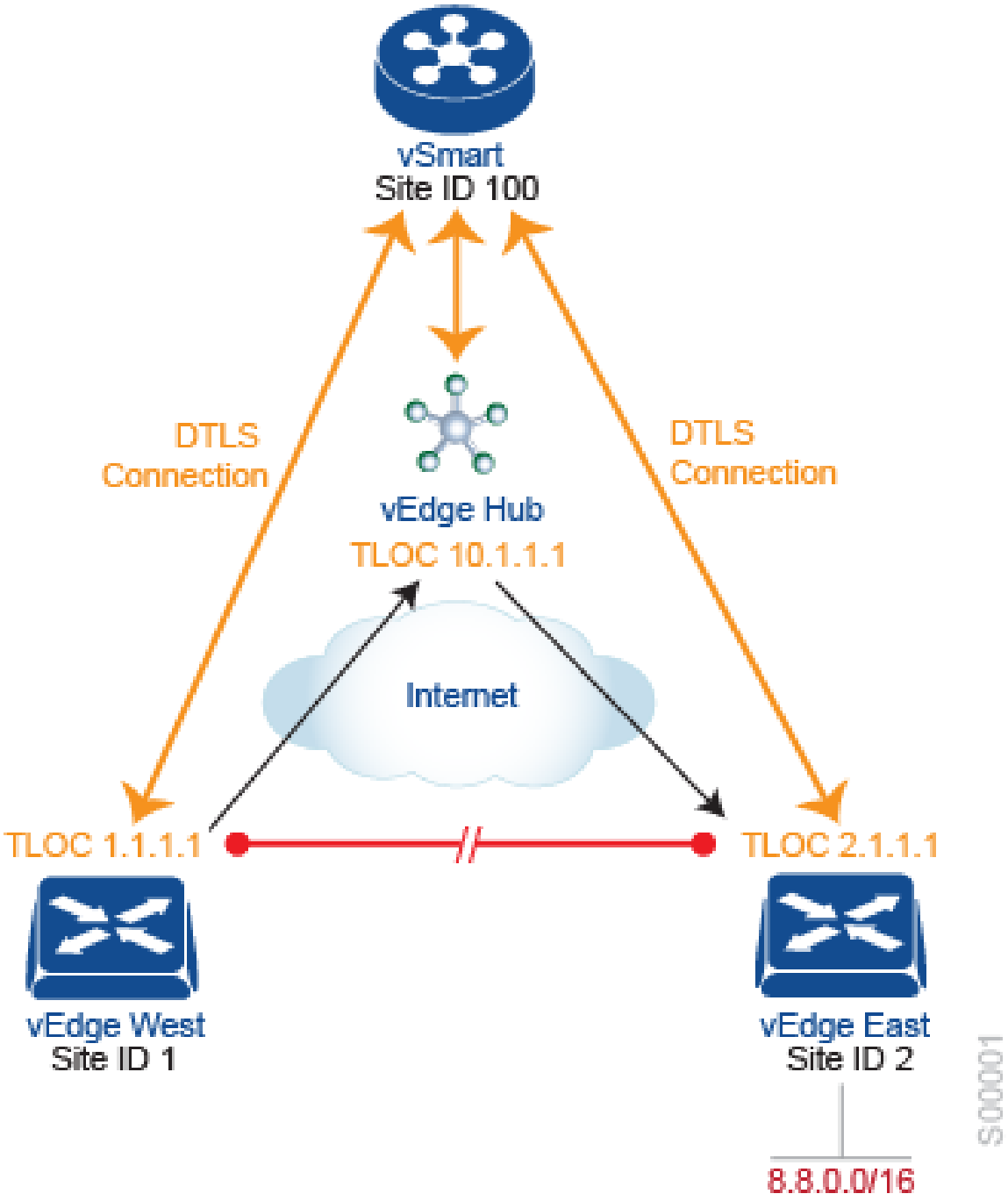
Examples of Modifying Traffic Flow with Centralized Control Policy

This section provides some basic examples of how you can use centralized control policies to modify the flow of data traffic through the overlay network.

Create an Arbitrary Topology

When data traffic is exchanged between two vEdge routers, if you have provisioned no control policy, the two vEdge routers establish an IPsec tunnel between them and the data traffic flows directly from one router to the next. For a network with only two vEdge routers or with just a small number of vEdge routers, establishing connections between each pair of routers is generally not be an issue. However, such a solution does not scale. In a network with hundreds or even thousands of branches, establishing a full mesh of IPsec tunnels would

tax the CPU resources of each vEdge router.



One way to minimize this overhead is to create a hub-and-spoke type of topology in which one of the vEdge routers acts as a hub site that receives the data traffic from all the spoke, or branch, routers and then redirects the traffic to the proper destination. This example shows one of the ways to create such a hub-and-spoke topology, which is to create a control policy that changes the address of the TLOC associated with the destination.

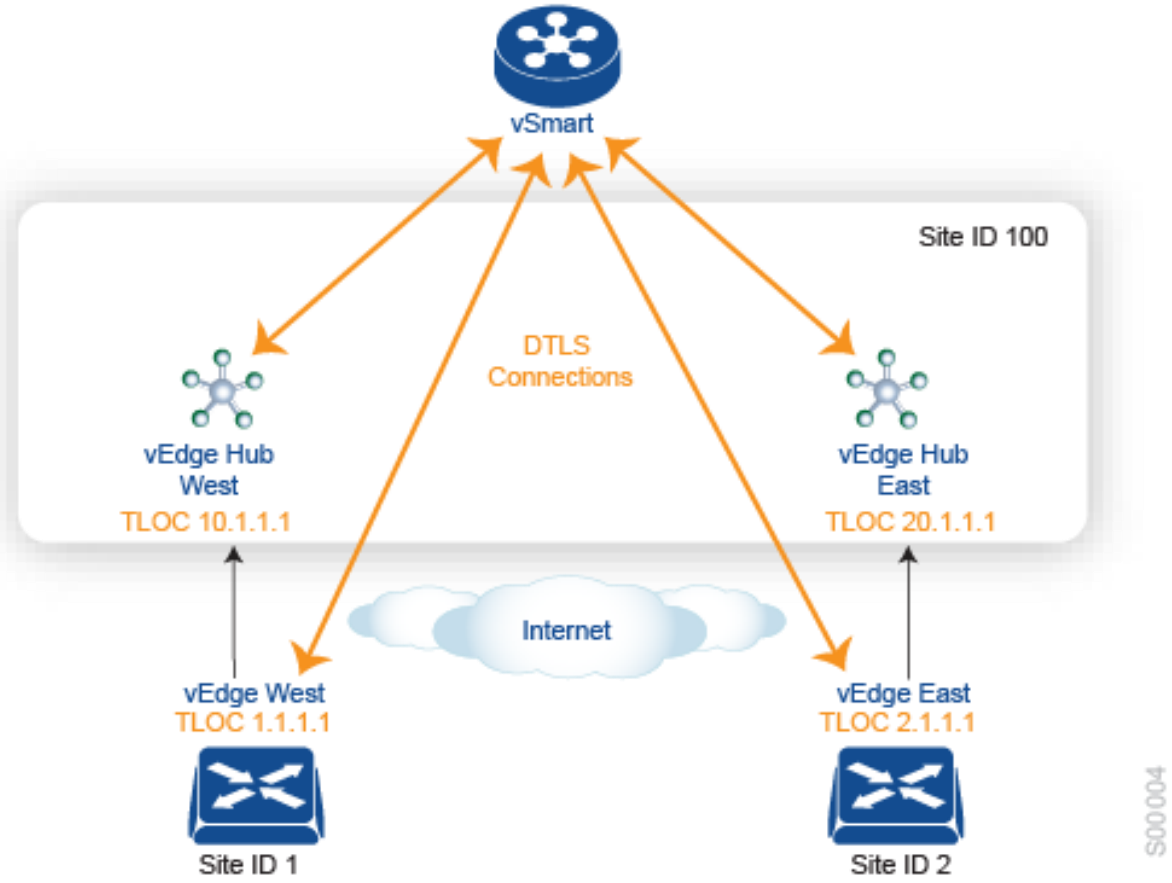
The figure here illustrates how such a policy might work. The topology has two branch locations, West and East. When no control policy is provisioned, these two vEdge routers exchange data traffic with each other directly by creating an IPsec tunnel between them (shown by the red line). Here, the route table on the West vEdge router contains a route to vEdge East with a destination TLOC of 2.1.1.1, color gold

(which we write as the tuple {2.1.1.1, gold}), and vEdge East route table has a route to the West branch with a destination TLOC of {1.1.1.1, gold}.

To set up a hub-and-spoke–type topology here, we provision a control policy that causes the West and East vEdge routers to send all data packets destined for the other router to the vEdge hub router. (Remember that because control policy is always centralized, you provision it on the vSmart controller.) On the West vEdge router, the policy simply changes the destination TLOC from {2.1.1.1, gold} to {10.1.1.1, gold}, which is the TLOC of the vEdge hub router, and on the East router, the policy changes the destination TLOC from {1.1.1.1, gold} to the hub's TLOC, {10.1.1.1, gold}. If there were other branch sites on the west and east sides of the network that exchange data traffic, you could apply these same two control policies to have them redirect all their data traffic through the hub vEdge router.

Set Up Traffic Engineering

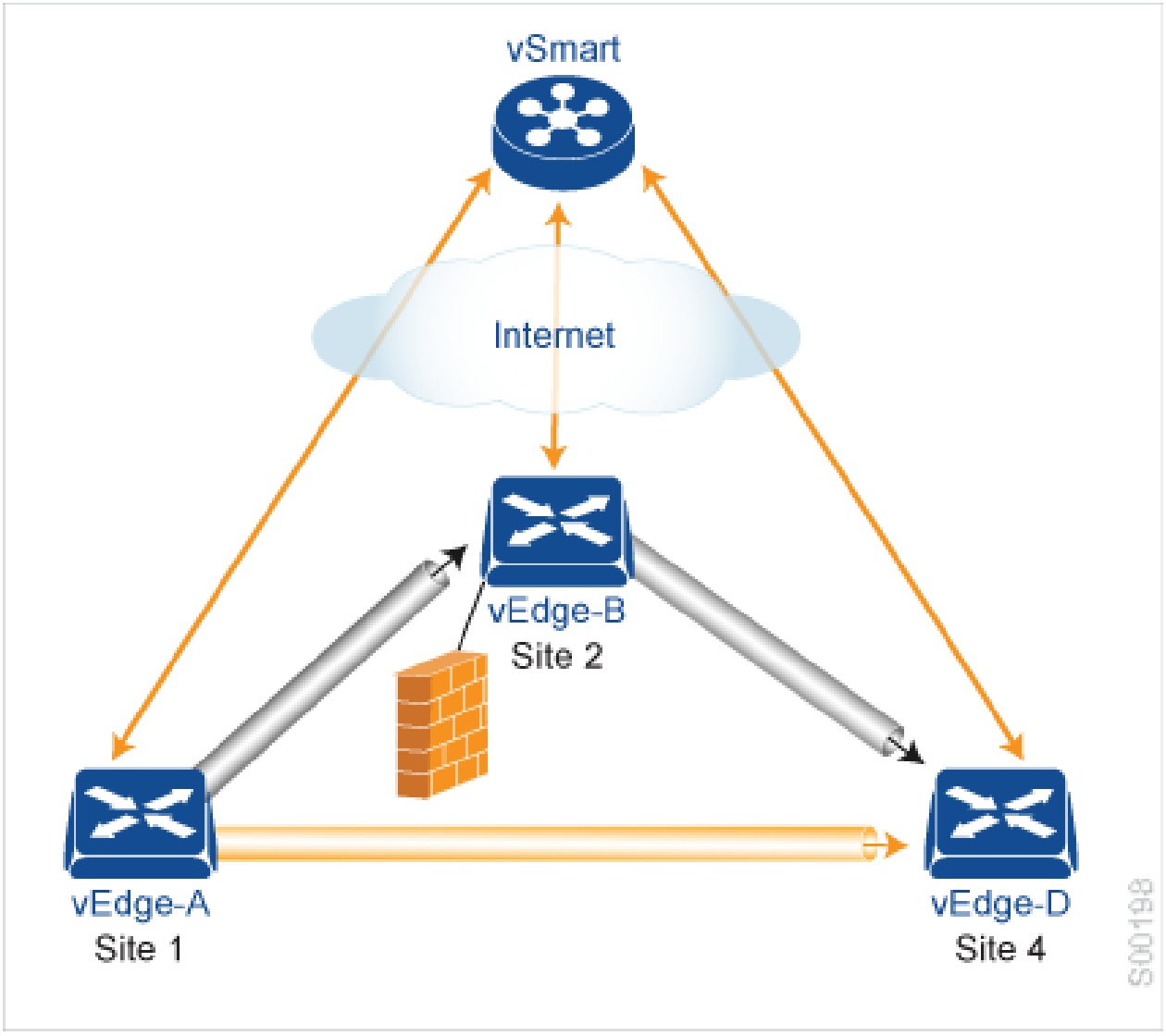
Control policy allows you to design and provision traffic engineering. In a simple case, suppose that you have two vEdge routers acting as hub devices. Here, you might want data traffic destined to a branch vEdge router to always transit through one of the hub vEdge routers. To engineer this traffic flow, you set the TLOC preference value to favor the desired hub vEdge router.



The figure on the left shows that Site ID 100 has two hub vEdge routers, one that serves the West side of the network and a second that serves the East side. We always want data traffic from the West vEdge branch router to be handled by the West vEdge hub, and similarly, we want data traffic from the East vEdge branch router to go through the east vEdge hub.

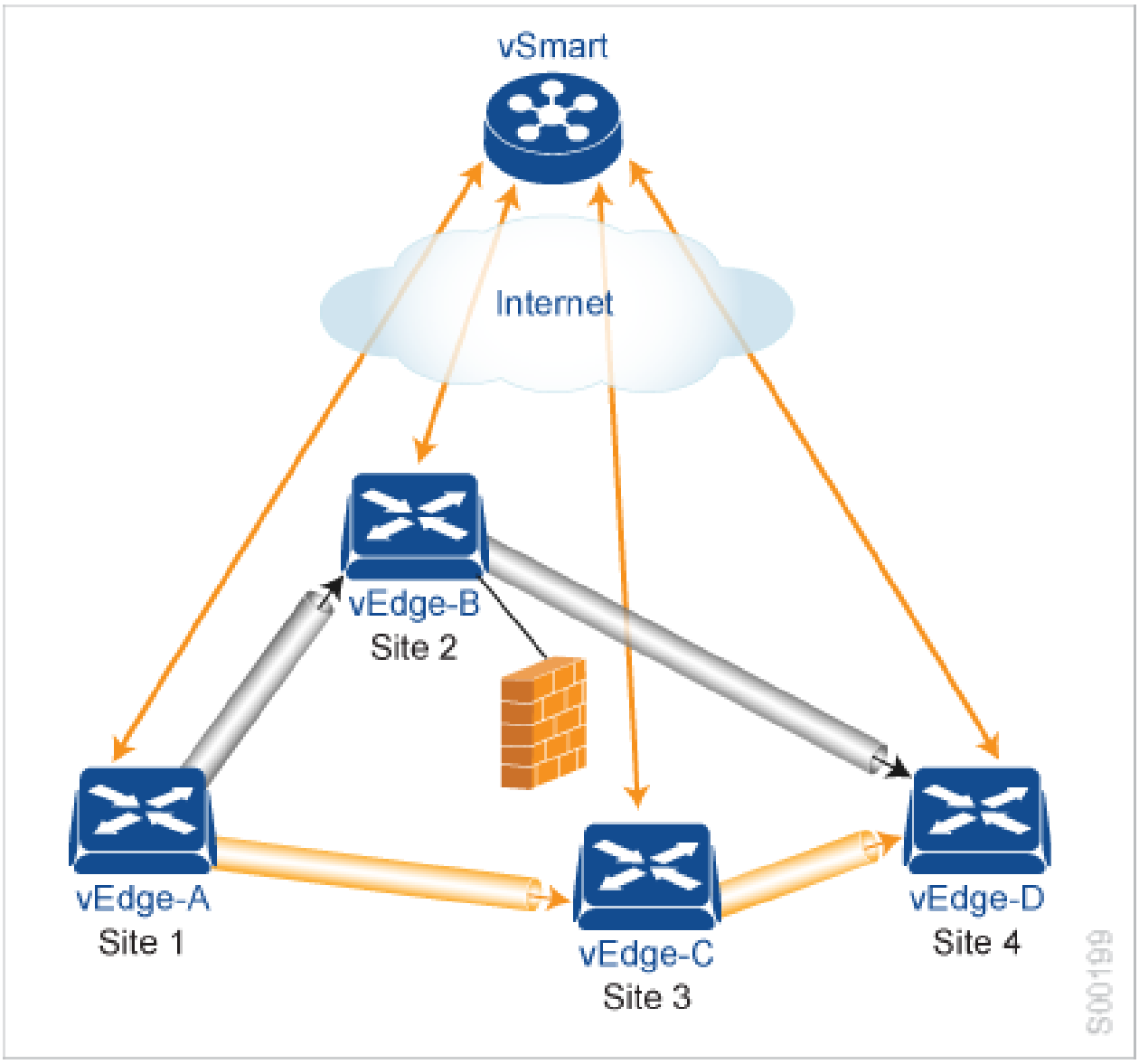
To engineer this traffic flow, you provision two control policies, one for Site ID 1, where the West vEdge branch router is located, and a second one for Site ID 2. The control policy for Site ID 1 changes the TLOC for traffic destined to the East vEdge router to {10.1.1.1, gold}, and the control policy for Site ID 2 changes the TLOC for traffic destined for Site ID 1 to {20.1.1.1, gold}. One additional effect of this traffic engineering policy is that it load-balances the traffic traveling through the two vEdge hub routers.

With such a traffic engineering policy, a route from the source router to the destination router is installed in the local route table, and traffic is sent to the destination regardless of whether the path between the source and destination vEdge routers is available. Enabling end-to-end tracking of the path to the ultimate destination allows the vSmart controller to monitor the path from the source to the destination, and to inform the source router when that path is not available. The source router can then modify or remove the path from its route table.



The figure to the right illustrates end-to-end path tracking. It shows that traffic from vEdge-A that is destined for vEdge-D first goes to an intermediate router, vEdge-B, perhaps because this intermediate router provides a service, such as a firewall. (You configure this traffic engineering with a centralized control policy that is applied to vEdge-A, at Site 1.) Then vEdge-B, which has a direct path to the ultimate destination, forwards the traffic to vEdge-D. So, in this example, the end-to-end path between vEdge-A and vEdge-D comprises two tunnels, one between vEdge-A and vEdge-B, and the second between vEdge-B and vEdge-D. The vSmart controller tracks this end-to-end path, and it notifies vEdge-A if the portion of the path between vEdge-B and vEdge-D becomes unavailable.

As part of end-to-end path tracking, you can specify how to forwarded traffic from the source to the ultimate destination via an intermediate router. (You do this by setting the TLOC action in the action portion of the control policy.) The default method is strict forwarding, where traffic is always sent from vEdge-A to vEdge-B, regardless of whether vEdge-B has a direct path to vEdge-D or whether the tunnel between vEdge-B and vEdge-D is up. More flexible methods forward some or all traffic directly from vEdge-A to vEdge-D. You can also set up a second intermediate router to provide a redundant path with the first intermediate router is unreachable and use an ECMP method to forward traffic between the two. The figure below adds vEdge-C as a redundant intermediate router.



Additional Information

Centralized Control Policy Configuration Examples
Configuring Centralized Control Policy
Policy Overview
Service Chaining

Configuring Centralized Control Policy

Centralized control policy, which you configure on vSmart controllers, affects routing policy based on information in OMP routes and OMP TLOCs. This type of policy allows you to set actions for matching routes and TLOCs, including redirecting packets through network services, such as firewalls, a feature that is called service chaining.

In domains with multiple vSmart controllers, all the controllers must have the same centralized control policy configuration to ensure that routing within the overlay network remains stable and predictable.

This article provides procedures for configuring centralized control policy (including service chaining) from the CLI.

Configuration Components

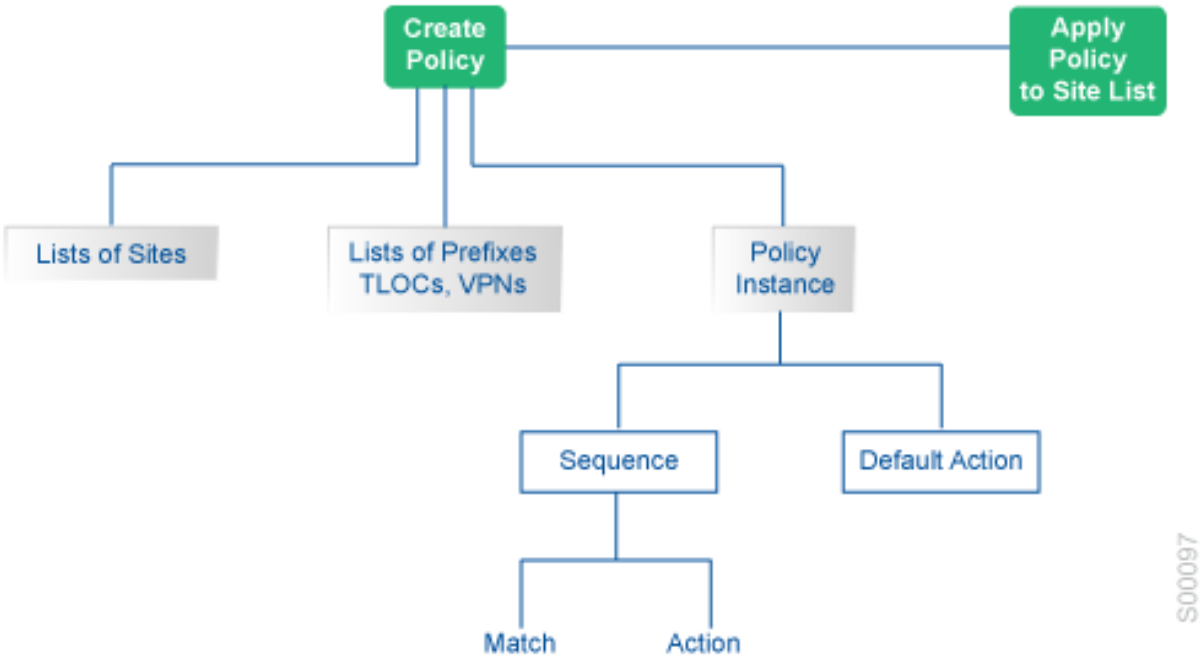
A centralized control policy consists of a series of numbered (ordered) sequences of match-action pair that are evaluated in order, from lowest sequence number to highest sequence number. When a route or TLOC matches one of the match conditions, the associated action is taken and policy evaluation on that packets stops. Keep this in mind as you design your policies to ensure that the desired actions are taken on the items subject to policy.

If a route or TLOC matches no parameters in any of the sequences in the policy configure, it is, by default, rejected and discarded.

To create a centralized control policy, you include the following components in the configuration on a vSmart controller:

Component	Description	Configuration Command
Lists	Groupings of related items that you reference in the match and action portions of the control policy configuration. The items you can group include overlay IP prefixes, network site IDs, TLOCs, and VPNs.	**policy lists**
Centralized control policy instance	Container for centralized control policy.	**policy control-policy**
Numbered sequences of match–action pairs	Sequences establish the order in which the policy components are applied.	**policy control-policy sequence**
Match parameters	Conditions that the routes and TLOCs must match to be considered for a control policy.	**policy control-policy sequence match route** — Match properties of OMP routes, including things such as the originating protocol and IP prefixes. **policy control-policy sequence match tloc** — Match transport location parameters, including things such as the domain ID and TLOC IP address.
Actions	Whether to accept or reject matching routes and TLOCs, and how to process matching items.	**policy control-policy sequence action**
Default action	Action to take if a route or TLOC matches none of the match parameters in any of the sequences. By default, nonmatching routes and TLOCs are rejected.	**policy control-policy default-action**
Application of centralized control policy	For a control policy to take effect, you apply it to one or more sites in the overlay network.	**apply-policy site-list control-policy**

The following figure illustrates the configuration components for centralized control policy.



General vManage Configuration Procedure

To configure centralized policies, use the vManage policy configuration wizard. The wizard consists of four sequential screens that guide you through the process of creating and editing policy components:

1. Create Applications or Groups of Interest—Create lists that group together related items and that you call in the match or action components of a policy.
2. Configure Topology—Create the network structure to which the policy applies.
3. Configure Traffic Rules—Create the match and action conditions of a policy.
4. Apply Policies to Sites and VPNs—Associate policy with sites and VPNs in the overlay network.

In the first three policy configuration wizard screens, you are creating policy components or blocks. In the last screen, you are applying policy blocks to sites and VPNs in the overlay network.

For a centralized policy to take effect, you must activate the policy.

Start the Policy Configuration Wizard

To start the policy configuration wizard:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy.

The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed.

Create Applications or Groups of Interest

To create lists of applications or groups to use in centralized control policy:

1. Start the policy configuration wizard as explained above.
2. Create new lists, as described in the following table:

1. List Type	Procedure
Color	1. In the left bar, click Color. 2. Click New Color List. 3. Enter a name for the list. 4. From the Select Color drop-down, select the desired colors. 5. Click Add.
Prefix	6. In the left bar, click Prefix. 7. Click New Prefix List. 8. Enter a name for the list. 9. In the Add Prefix field, enter one or more data prefixes separated by commas. 10. Click Add.
Site	11. In the left bar, click Site. 12. Click New Site List. 13. Enter a name for the list. 14. In the Add Site field, enter one or more site IDs separated by commas. 15. Click Add.
TLOC	16. In the left bar, click TLOC. 17. Click New TLOC List. The TLOC List popup displays. 18. Enter a name for the list. 19. In the TLOC IP field, enter the system IP address for the TLOC. 20. In the Color field, select the TLOC's color. 21. In the Encap field, select the encapsulation type. 22. In the Preference field, optionally select a preference to associate with the TLOC. 23. Click Add TLOC to add another TLOC to the list. 24. Click Save.

VPN	25. In the left bar, click VPN. 26. Click New VPN List. 27. Enter a name for the list. 28. In the Add VPN field, enter one or more VPN IDs separated by commas. 29. Click Add.

4. Click Next to move to Configure Topology in the wizard. When you first open this screen, the Topology tab is selected by default.

Configure the Network Topology

To configure the network topology to use in centralized control policy:

1. If you are already in the policy configuration wizard, skip to Step 4. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Create a network topology, as described in the following table:

Policy Type	Description	Procedure

Custom Control (Route & TLOC)	Centralized route control policy (for matching OMP routes)	1. In the Add Topology drop-down, select Custom Control (Route & TLOC). 2. Enter a name for the control policy. 3. Enter a description for the policy. 4. In the left pane, click Add Sequence Type. The Add Control Policy popup displays. 5. Select Route. A policy component containing the text string Route is added in the left pane. 6. Double-click the Route text string, and enter a name for the policy component. 7. In the right pane, click Add Sequence Rule. The Match/Actions box opens, and Match is selected by default. 8. From the boxes under the Match box, select the desired policy match type. Then select or enter the value for that match condition. Configure additional match conditions for the sequence rule, as desired. For an explanation of the match conditions, see the OMP Route Match Attributes section in the Configuring Centralized Control Policy article for your software release. 9. Click Actions. The Reject radio button is selected by default. To configure actions to perform on accepted packets, click the Accept radio button. Then select the action or enter a value for the action. For an explanation of the actions, see the Action Parameters section in the Configuring Centralized Control Policy article for your software release. 10. Click Save Match and Actions. 11. Click Add Sequence Rules to configure more sequence rules, as desired. Drag and drop to re-order them. 12. Click Add Sequence Type to configure more sequences, as desired. Drag and drop to re-order them. 13. Click Save Control Policy.

	Centralized TLOC control policy (for matching TLOC routes)	14. In the Add Topology drop-down, select Custom Control (Route & TLOC). 15. Enter a name for the control policy. 16. Enter a description for the policy. 17. In the left pane, click Add Sequence Type. The Add Control Policy popup displays. 18. Select TLOC. A policy component containing the text string TLOC is added in the left pane. 19. Double-click the TLOC text string, and enter a name for the policy component. 20. In the right pane, click Add Sequence Rule. The Match/Actions box opens, and Match is selected by default. 21. From the boxes under the Match box, select the desired policy match type. Then select or enter the value for that match condition. Configure additional match conditions for the sequence rule, as desired. For an explanation of the match conditions, see the OMP TLOC Match Attributes section in the Configuring Centralized Control Policy article for your software release. 22. Click Actions. The Reject radio button is selected by default. To configure actions to perform on accepted packets, click the Accept radio button. Then select the action or enter a value for the action. For an explanation of the actions, see the Action Parameters section in the Configuring Centralized Control Policy article for your software release. 23. Click Save Match and Actions. 24. Click Add Sequence Rules to configure more sequence rules, as desired. Drag and drop to re-order them. 25. Click Add Sequence Type to configure more sequences, as desired. Drag and drop to re-order them. 26. Click Save Control Policy.

5. To use an existing topology:
    a. In the Add Topology drop-down, click Import Existing Topology. The Import Existing Topology popup displays.
    b. Select the type of topology.
    c. In the Policy drop-down, select the name of the topology.
    d. Click Import.
6. Click Next to move to Configure Traffic Rules in the wizard.
7. Click Next to move to Apply Policies to Sites and VPNs in the wizard.

Apply Policies to Sites and VPNs

In the last screen of the policy configuration wizard, you associate the policy blocks that you created on the previous three screens with VPNs and with sites in the overlay network.

To apply a policy block to sites and VPNs in the overlay network:

1. If you are already in the policy configuration wizard, skip to Step 6. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed

3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.
5. Click Next. The Apply Policies to Sites and VPNs screen opens.
6. In the Policy Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
7. In the Policy Description field, enter a description of the policy. It can contain up to 2048 characters. This field is mandatory, and it can contain any characters and spaces.
8. From the Topology bar, select the type of policy block. The table then lists policies that you have created for that type of policy block.
9. Associate the policy with VPNs and sites. The choice of VPNs and sites depends on the type of policy block:
    a. For a Topology policy block, click Add New Site List and VPN List or Add New Site. Some topology blocks might have no Add buttons. Select one or more site lists, and select one or more VPN lists. Click Add.
    b. For an Application-Aware Routing policy block, click Add New Site List and VPN list. Select one or more site lists, and select one or more VPN lists. Click Add.
    c. For a Traffic Data policy block, click Add New Site List and VPN List. Select the direction for applying the policy (From Tunnel, From Service, or All), select one or more site lists, and select one or more VPN lists. Click Add.
    d. For a cflowd policy block, click Add New Site List. Select one or more site lists, Click Add.
10. Click Preview to view the configured policy. The policy is displayed in CLI format.
11. Click Save Policy. The Configuration ► Policies screen opens, and the policies table includes the newly created policy.

Activate a Centralized Policy

Activating a centralized policy sends that policy to all connected vSmart controllers. To activate a centralized policy:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Select a policy.
3. Click the More Actions icon to the right of the row, and click Activate. The Activate Policy popup opens. It lists the IP addresses of the reachable vSmart controllers to which the policy is to be applied.
4. Click Activate.

General CLI Configuration Procedure

To configure a centralized control policy using the CLI:

1. Create a list of overlay network sites to which the centralized control policy is to be applied (in the **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists** **site-list** *list-name*
   vSmart(config-lists- *list-name*)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).
   Create additional site lists, as needed.

2. Create lists of IP prefixes, TLOCs, and VPNs, as needed:
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **prefix-list** *list-name*
   vSmart(config-lists- *list-name*)# **ip-prefix** *prefix* / *length*

   vSmart(config)# **policy lists**
   vSmart(config-lists)# **tloc-list** *list-name*
   vSmart(config-lists- *list-name*)# **tloc** *address* **color** *color* **encap** *encapsulation* [**preference** *value*]

   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists- *list-name*)# **vpn** *vpn-id*

3. Create a control policy instance:
   vSmart(config)# **policy control-policy** *policy-name*
   vSmart(config-control-policy- *policy-name*)#

4. Create a series of match–action pair sequences:
   vSmart(config-control-policy- *policy-name*)# **sequence** *number*
   vSmart(config-sequence- *number*)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

5. Define match parameters for routes and for TLOCs:
   vSmart(config-sequence- *number*)# **match route** *route-parameter*
   vSmart(config-sequence- *number*)# **match tloc** *tloc-parameter*

6. Define actions to take when a match occurs:
   vSmart(config-sequence- *number*)# **action reject**
   vSmart(config-sequence- *number*)# **action accept export-to** (**vpn** *vpn-id* | **vpn-list** *list-name*)
   vSmart(config-sequence- *number*)# **action accept set omp-tag** *number*
   vSmart(config-sequence- *number*)# **action accept set preference** *value*
   vSmart(config-sequence- *number*)# **action accept set service** *service-name* (**tloc** *ip-address* | **tloc-list** *list-name*) [**vpn** *vpn-id*]
   vSmart(config-sequence- *number*)# **action accept set tloc** *ip-address* **color** *color* [**encap** *encapsulation*]
   vSmart(config-sequence- *number*)# **action accept set tloc-action** *action*
   vSmart(config-sequence- *number*)# **action accept set tloc-list** *list-name*

7. Create additional numbered sequences of match–action pairs within the control policy, as needed.

8. If a route does not match any of the conditions in one of the sequences, it is rejected by default. If you want nonmatching routes to be accepted, configure the default action for the policy:
   vSmart(config- *policy-name*)# **default-action accept**

9. Apply the policy to one or more sites in the Viptela overlay network:
   vSmart(config)# **apply-policy site-list** *list-name* **control-policy** *policy-name* (**in** | **out**)

10. If the action you are configuring is a service, configure the required services on the vEdge routers so that the vSmart controller knows how to reach the services:
    vEdge(config)# **vpn** *vpn-id* **service** *service-name* **address** *ip-address*
    Specify the VPN is which the service is located and one to four IP addresses to reach the service device or devices. If multiple devices provide the same service, the vEdge router load-balances the traffic among them. Note that the vEdge router keeps track of the services, advertising them to the vSmart controller only if the address (or one of the addresses) can be resolved locally, that is, at the vEdge router's local site, and not learned through OMP. If a previously advertised service becomes unavailable, the vEdge router withdraws the service advertisement.

Structural Components of Policy Configuration for Centralized Control Policy

Following are the structural components required to configure centralized control policy. Each one is explained in more detail in the sections below.

```
policy
  lists
    color-list list-name
      color color
    prefix-list list-name
      ip-prefix prefix
    site-list list-name
      site-id site-id
    tloc-list list-name
      tloc address color color encap encapsulation [preference value]
    vpn-list list-name
      vpn vpn-id
  control-policy policy-name
    sequence number
      match
        match-parameters
      action
        reject
        accept
          export-to vpn
        accept
          set parameter
    default-action
      (accept | reject)
apply-policy site-list list-name
  control-policy policy-name (in | out)
```

Lists

Centralized control policy uses the following types of lists to group related items. You configure lists under the **policy lists** command hierarchy.

List Type	Description	Command
Color list	List of one or more TLOC colors. *color* can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** through **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . To configure multiple colors in a single list, include multiple **color** options, specifying one color in each option.	**color-list** *list-name* **color** *color*
Prefix list	List of one or more IP prefixes. Specify the IP prefixes as follows: • *prefix* / *length* —Exactly match a single prefix–length pair. • **0.0.0.0/0** —Match any prefix–length pair. • **0.0.0.0/0 le** *length* —Match any IP prefix whose length is less than or equal to *length* . For example, **ip-prefix 0.0.0.0/0 le 16** matches all IP prefixes with lengths from /1 through /16. • **0.0.0.0/0 ge** *length* —Match any IP prefix whose length is greater than or equal to *length* . For example, **ip-prefix 0.0.0.0 ge 25** matches all IP prefixes with lengths from /25 through /32. • **0.0.0.0/0 ge** *length1* **le** *length2* , or **0.0.0.0 le** *length2* **ge** *length1* —Match any IP prefix whose length is greater than or equal to *length1* and less than or equal to *length2* . For example, **ip-prefix 0.0.0.0/0 ge 20 le 24** matches all /20, /21, /22, /23, and /24 prefixes. Also, **ip-prefix 0.0.0.0/0 le 24 ge 20** matches the same prefixes. If *length1* and *length2* are the same, a single IP prefix length is matched. For example, **ip-prefix 0.0.0.0/0 ge 24 le 24** matches only /24 prefixes. To configure multiple prefixes in a single list, include multiple **ip-prefix** options, specifying one prefix in each option.	**prefix-list** *list-name* **ip-prefix** *prefix* / *length*

Site list	List of one of more site identifiers in the overlay network. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**). To configure multiple sites in a single list, include multiple **site-id** options, specifying one site number in each option.	**site-list** *list-name* **site-id** *site-id*
TLOC list	List of one or more transport locations (TLOCs). For each TLOC you must specify its IP address, color, and encapsulation. The IP address must be the system IP address. Setting a preference value is optional. To configure multiple TLOCs in a single list, include multiple **tloc** options, specifying one TLOC number in each option.	**tloc-list** *list-name* **tloc** *ip-address* **color** *color* **encap** (**gre** | **ipsec**) [**preference** *number*]
VPN list	List of one or more VPNs in the overlay network. You can specify a single VPN identifier (such as **vpn-id 1**) or a range of VPN identifiers (such as **vpn-id 1-10**). To configure multiple VPNs in a single list, include multiple **vpn** options, specifying one VPN number in each option.	**vpn-list** *list-name* **vpn** *vpn-id*

Sequences

A centralized control policy contains sequences of match–action pairs. The sequences are numbered to set the order in which a route or TLOC is analyzed by the match–action pairs in the policy. You configure sequences with the **policy control-policy sequence** command.

Each sequence in a centralized control policy can contain one **match** command (either **match route** or **match tloc**) and one **action** command.

Match Parameters

Centralized control policy can match OMP route (vRoute) or TLOC route attributes. You configure the OMP route attributes to match with the **policy control-policy sequence match route** command, and you configure the TLOC attributes to match with the **policy control-policy sequence match tloc** command.

Each sequence in a policy can contain one **match** section—either **match route** or **match tloc** .

OMP Route Match Attributes

For OMP routes (vRoutes), you can match these attributes:

Description	Command	Value or Range
Individual color.	**color** *color*	**3g** , **biz-internet** , **blue** , **bronze** , **custom1** through **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver**
One or more colors.	**color-list** *list-name*	Name of a **policy lists color-list** list.
Tag value associated with the route or prefix in the routing database on the vEdge router.	**omp-tag** *number*	0 through 4294967295
Protocol from which the route was learned.	**origin** *protocol*	**bgp-external** , **bgp-internal** , **connected** , **ospf-external1** , **ospf-external2** , **ospf-inter-area** , **ospf-intra-area** , **static**

IP address from which the route was learned.	**originator** *ip-address*	IP address
How preferred a prefix is. This is the preference value that the route or prefix has in the local site, that is, in the routing database on the vEdge router. A higher preference value is more preferred.	**preference** *number*	0 through 255
One or more prefixes.	**prefix-list** *list-name*	Name of a **policy lists prefix-list** list.
Individual site identifier.	**site-id** *site-id*	0 through 4294967295
One or more overlay network site identifiers.	**site-list** *list-name*	Name of a **policy lists site-list** list.
Individual TLOC address.	**tloc** *ip-address*	IP address
One or more TLOC addresses.	**tloc-list** *list-name*	Name of a **policy lists tloc-list** list.
Individual VPN identifier.	**vpn** *vpn-id*	0 through 65535
One or more VPN identifiers.	**vpn-list** *list-name*	Name of a **policy lists vpn-list** list.

TLOC Route Match Attributes

For TLOC routes, you can match these attributes:

Description	Command	Value or Range
Carrier for the control traffic.	**carrier** *carrier-name*	**default** , **carrier1** through **carrier8**
Individual color.	**color** *color*	**3g** , **biz-internet** , **blue** , **bronze** , **custom1** through **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver**
One or more colors.	**color-list** *list-name*	See the colors above.
Domain identifier associated with a TLOC.	**domain-id** *domain-id*	0 through 4294967295
Tag value associated with the TLOC route in the route table on the vEdge router.	**omp-tag** *number*	0 through 4294967295
IP address from which the route was learned.	**originator** *ip-address*	IP address
How preferred a TLOC route is. This is the preference value that the TLOC route has in the local site, that is, in the route table on the vEdge router. A higher preference value is more preferred.	**preference** *number*	0 through 255

Individual site identifier.	**site-id** *site-id*	0 through 4294967295
One or more overlay network site identifiers.	**site-list** *list-name*	Name of a **policy lists site-list** list.
Individual TLOC address.	**tloc** *address*	IP address
One or more TLOC addresses.	**tloc-list** *list-name*	Name of a **policy lists tloc-list** list.

Action Parameters

For each match condition, you configure a corresponding action to take if the route or TLOC matches. You configure this with the **policy control-policy action** command.

Each sequence in a centralized control policy can contain one **action** command.

In the action, you first specify whether to accept or reject a matching route or TLOC:

Description	Parameter	Value or Range
Accept the route. An accepted route is eligible to be modified by the additional parameters configured in the **action** portion of the policy configuration.	**accept**	—
Discard the packet.	**reject**	—

Then, for a route or TLOC that is accepted, you can configure the following actions:

Description	Parameter	Value or Range
Export the route the the specified VPN or list of VPNs (for a **match route** match condition only).	**export-to** (**vpn** *vpn-id* | **vpn-list** *vpn-list*)	0 through 65535 or list name.
Change the tag string in the route, prefix, or TLOC.	**set omp-tag** *number*	0 through 4294967295
Change the preference value in the route, prefix, or TLOC to the specified value. A higher preference value is more preferred.	**set preference** *number*	0 through 255
Specify a service to redirect traffic to before delivering the traffic to its destination. The TLOC address or list of TLOCs identifies the TLOCs to which the traffic should be redirected to reach the service. In the case of multiple TLOCs, the traffic is load-balanced among them. The VPN identifier is where the service is located. Configure the services themselves on the vEdge routers that are collocated with the service devices, using the **vpn service** configuration command.	**set service** *service-name* (**tloc** *ip-address* | **tloc-list** *list-name*) [**vpn** *vpn-id*]	Standard services: **FW** , **IDS** , **IDP** Custom services: **netsvc1** , **netsvc2** , **netsvc3** , **netsvc4** TLOC list configured with a **policy lists tloc-list** command.

Change the TLOC address, color, and encapsulation to the specified address and color.	**set tloc** *ip-address* **color** *color* [**encap** *encapsulation*]	IP address, TLOC color, and encapsulation, Color can be one of **3g** , **biz-internet** , **blue** , **bronze** , **custom1** through **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . Encapsuation can be either **gre** or **ipsec** .
Direct matching routes or TLOCs using the mechanism specified by *action* , and enable end-to-end tracking of whether the ultimate destination is reachable. Setting a TLOC action is useful when traffic is first directed, via policy, to an intermediate destination, which then forwards the traffic to its ultimate destination. For example, for traffic from vEdge-A destined for vEdge-D, a policy might direct traffic from vEdge-A first to vEdge-B (the intermediate destination), and vEdge-B then sends it to the final destination, vEdge-D. Setting the TLOC action option enables the vSmart controller to perform end-to-end tracking of the path to the ultimate destination router. In our example, matching traffic goes from vEdge-A to vEdge-B and then, in a single hop, goes to vEdge-D. If the tunnel between vEdge-B and vEdge-D goes down, the vSmart controller relays this information to vEdge-A, and vEdge-A removes its route to vEdge-D from its local route table. End-to-end tracking works here only because traffic goes from vEdge-B to vEdge-D in a single hop, via a single tunnel. If the traffic from vEdge-A went first to vEdge-B, then to vEdge-C, and finally to vEdge-D, the vSmart controller is unable to perform end-to-end tracking and is thus unable to keep vEdge-A informed about whether full path between it and vEdge-D is up.	**set tloc-action** *action*	**ecmp** —Equally direct matching control traffic between the intermediate destination and the ultimate destination. In our example, traffic would be sent to vEdge-B (which would then send it to vEdge-D) and directly to vEdge-D. With this action, if the intermediate destination is down, all traffic reaches the ultimate destination. **primary** —First direct matching traffic to the intermediate destination. If that router is not reachable, then direct it to the final destination. In our example, traffic would first be sent to vEdge-B. If this router is down, it is sent directly to vEdge-D. With this action, if the intermediate destination is down, all traffic reaches the final destination. **backup** —First direct matching traffic to the final destination. If that router is not reachable, then direct it to the intermediate destination. In our example, traffic would first be sent directly to vEdge-D. If the vEdge-A is not able to reach vEdge-D, traffic is sent to vEdge-B, which might have an operational path to reach vEdge-D. With this action, if the source is unable to reach the final destination directly, it is possible for all traffic to reach the final destination via the intermediate destination. **strict** —Direct matching traffic only to the intermediate destination. In our example, traffic is sent only to vEdge-B, regardless of whether it is reachable. With this action, if the intermediate destination is down, no traffic reaches the final destination. If you do not configure a set tloc-action action in a centralized control policy, strict is the default behavior.
Change the TLOC address and color to those in the specified TLOC list.	**set tloc-list** *list-name*	Name of a **policy lists tloc-list** list.

Default Action

If a route or TLOC being evaluated does not match any of the match conditions in a centralized control policy, a default action is applied to it. By default, the route or TLOC is rejected. To modify this behavior, include the **control policy default-action accept** command.

Applying Centralized Control Policy

For a centralized control policy to take effect, you apply it to a list of sites in the overlay network with the following command:

vSmart(config)# **apply-policy site-list** *list-name* **control-policy** *policy-name* (**in** | **out**)

You apply centralized control policy directionally:

- Inbound direction (**in**)—The policy analyzes routes and TLOCs being received from the sites in the site list before placing the routes and TLOCs into the route table on the vSmart controller, so the specified policy actions affect the OMP routes stored in the route table.

- Outbound direction (**out**)—The policy analyzes routes and TLOCs in the vSmart controller's route table after they are exported from the route table.

For all **control-policy** policies that you apply with **apply-policy** commands, the site IDs across all the site lists must be unique. That is, the site lists must not contain overlapping site IDs. An example of overlapping site IDs are those in the two site lists **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** . Here, sites 70 through 100 are in both lists. If you were to apply these two site lists to two different **control-policy** policies, the attempt to commit the configuration on the vSmart controller would fail.

The same type of restriction also applies to the following types of policies:

- Application-aware routing policy (**app-route-policy**)
- Centralized data policy (**data-policy**)
- Centralized data policy used for cflowd flow monitoring (**data-policy** hat includes a **cflowd** action and **apply-policy** that includes a **cflowd-template** command)

You can, however, have overlapping site IDs for site lists that you apply for different types of policy. For example, the sites lists for **control-policy** and **data-policy** policies can have overlapping site IDs. So for the two example site lists above, **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** , you could apply one to a control policy and the other to a data policy.

Additional Information

Centralized Control Policy
Centralized Control Policy Configuration Examples

Centralized Control Policy Configuration Examples

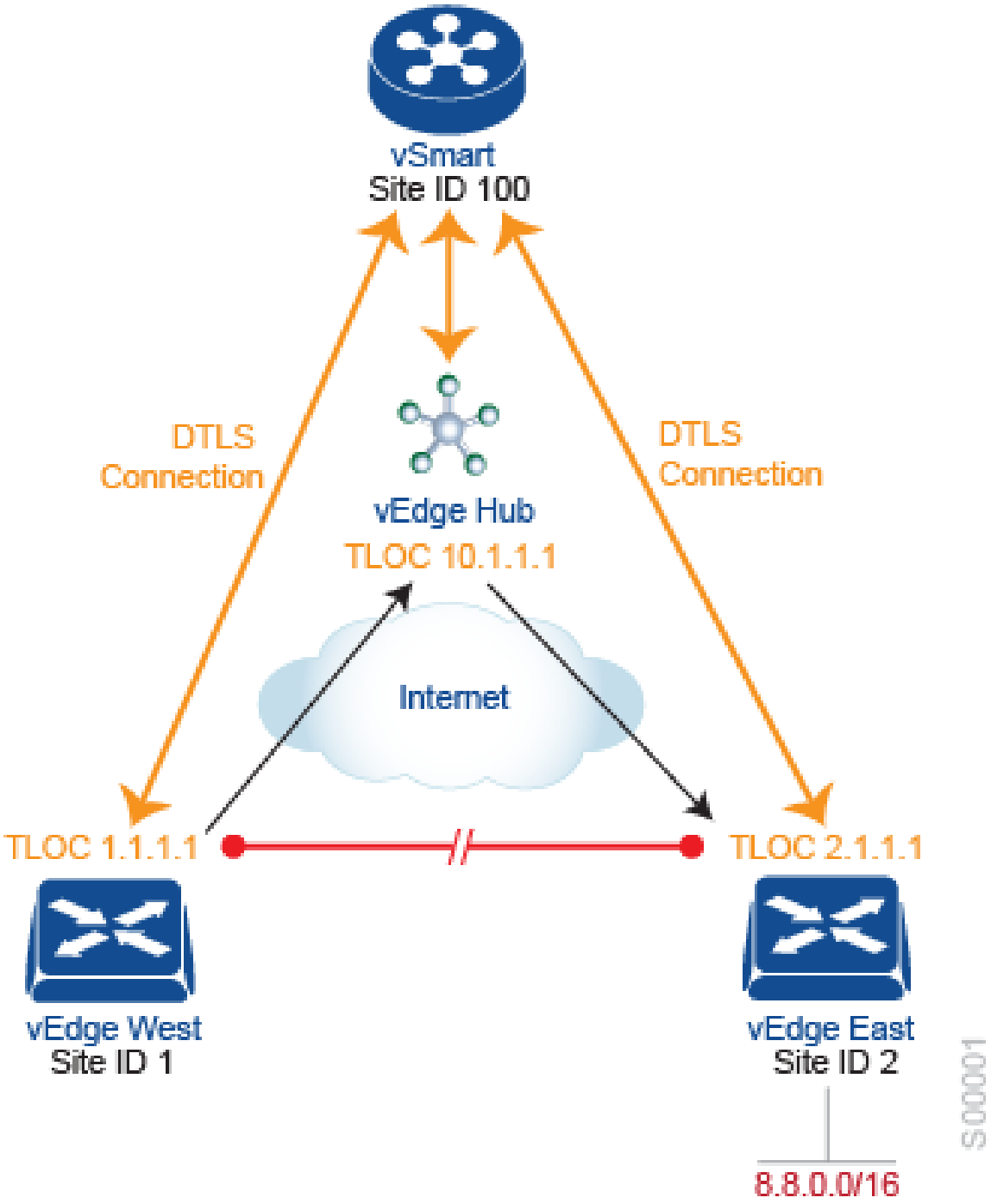
This article provides some straightforward examples of configuring centralized control policy to help you understand the configuration procedure and get an idea of how to use policy to influence traffic flow across the Viptela overlay network domain. You can find more complex examples in the **Validated Examples** articles.

Traffic Engineering

This example of traffic engineering forces all traffic to come to a vEdge router via a vEdge hub instead of directly.

One common way to design a domain in a Viptela overlay network is to route all traffic destined for branches through a vEdge hub router, which is typically located in a data center, rather than sending the traffic directly from one vEdge router to another. You can think of this as a hub-and-spoke design, where one vEdge router is acting as a hub and the vEdge routers are the spokes. With such a design, traffic between local branches travels over the IPsec connections that are established between the vEdge spoke routers and the vEdge hub routers when the vEdge routers booted up. Using established connections means that the vEdge routers do not need to expend time and CPU cycles to establish IPsec connections with each other. If you were to imagine that this were a large network with many vEdge routers, having a full mesh of connections between each pair of routers would require a large amount of CPU from the routers. Another attribute of this design is that, from an administrative point of view, it can be simpler to institute coordinated traffic flow policies on the vEdge hub routers, both because there are fewer of them in the overlay network and because they are located in a centralized data center.

One way to direct all vEdge spoke router traffic to a vEdge hub router is to create a policy that changes the TLOC associated with the routes in the local network. Let's consider the topology in the figure here:



This topology has two vEdge routers in different branches:

- The West vEdge router in site ID 1. The TLOC for this router is defined by its IP address (1.1.1.1), a color (gold), and an encapsulation (here, IPsec). We write the full TLOC address as {1.1.1.1, gold, ipsec}. The color is simply a way to identify a flow of traffic and to separate it from other flows.

- The East vEdge router in site ID 2 has a TLOC address of {2.1.1.1, gold, ipsec}.

The West and East vEdge routers learn each other's TLOC addresses from the OMP routes distributed to them by the vSmart controller. In this example, the East vEdge advertises the prefix 8.8.0.0/16 as being reachable at TLOC {2.1.1.1, gold, ipsec}. In the absence of any policy, the West vEdge router could route traffic destined for 8.8.0.0/16 to TLOC {2.1.1.1, gold, ipsec}, which means that the West vEdge router would be sending traffic directly to the East vEdge router.

However, our design requires that all traffic from West to East be routed through the vEdge hub router, whose TLOC address is {100.1.1.1, gold, ipsec}, before going to the East vEdge router. To effect this traffic flow, you define a policy that changes the route's TLOC. So, for the prefix 8.8.0.0/16, you create a policy that changes the TLOC associated with the prefix 8.8.0.0/16 from {2.1.1.1, gold, ipsec}, which is the TLOC address of the East vEdge router, to {10.1.1.1, gold, ipsec}, which is the TLOC address of the vEdge hub router. The result is that the OMP route for the prefix 8.8.0.0/16 that the vSmart controller advertises to the West vEdge router contains the TLOC address of the vEdge hub router instead of the TLOC address of the East vEdge router. From a traffic flow point of view, the West vEdge router then sends all traffic destined for 8.8.0.0/16 to the vEdge hub router.

The vEdge Hub router also learns the TLOC addresses of the West and East vEdge routers from the OMP routes advertised by the vSmart controller. Because we want vEdge hub router to use these two TLOC addresses, no policy is required to control how the hub directs traffic to the the vEdge routers.

Here is a policy configuration on the vSmart controller that directs the West vEdge router (and any other vEdges in the network domain) to send traffic destined to prefix 8.8.0.0/16 to TLOC 10.1.1.1, gold, which is the vEdge hub router:

```
policy
  lists
    prefix-list east-prefixes
      ip-prefix 8.8.0.0/16
    site-list west-sites
      site-id 1
  control-policy change-tloc
    sequence 10
      match route
        prefix-list east-prefixes
        site-id 2
      action accept
        set tloc 10.1.1.1 color gold encap ipsec
apply-policy
  site west-sites control-policy change-tloc out
```

A rough English translation of this policy is:

```
Create a list named "east-prefixes" that contains the IP prefix "8.8.0.0/16"
  Create a list named "west-sites" that contains the site-id "1"
  Define a control policy named "change-tloc"
    Create a policy sequence element that:
      Matches a prefix from list "east-prefixes", that is, matches "8.8.0.0/16"
      AND matches a route from site-id "2"
    If a match occurs:
      Accept the route
      AND change the route's TLOC to "10.1.1.1" with a color of "gold" and an encapsulation of "ipsec"
  Apply the control policy "change-tloc" to OMP routes sent by the vSmart
    controller to "west-sites", that is, to site ID 1
```

This control policy is configured on the vSmart controller as an outbound policy, as indicated by the **out** option in the **apply-policy site** command. This option means the the vSmart controller applies the TLOC change to the OMP route after it distributes the route from its route table. The OMP route for prefix 8.8.0.0/16 that the vSmart controller distributes to the West vEdge associates 8.8.0.0/16 with TLOC 10.1.1.1, gold. This is the OMP route that the West vEdge router installs it in its route table. The end results are that when the West vEdge router sends traffic to 8.8.0.0/16, the traffic is directed to the vEdge hub; and the West vEdge router does not establish a DTLS tunnel directly with the East vEdge router.

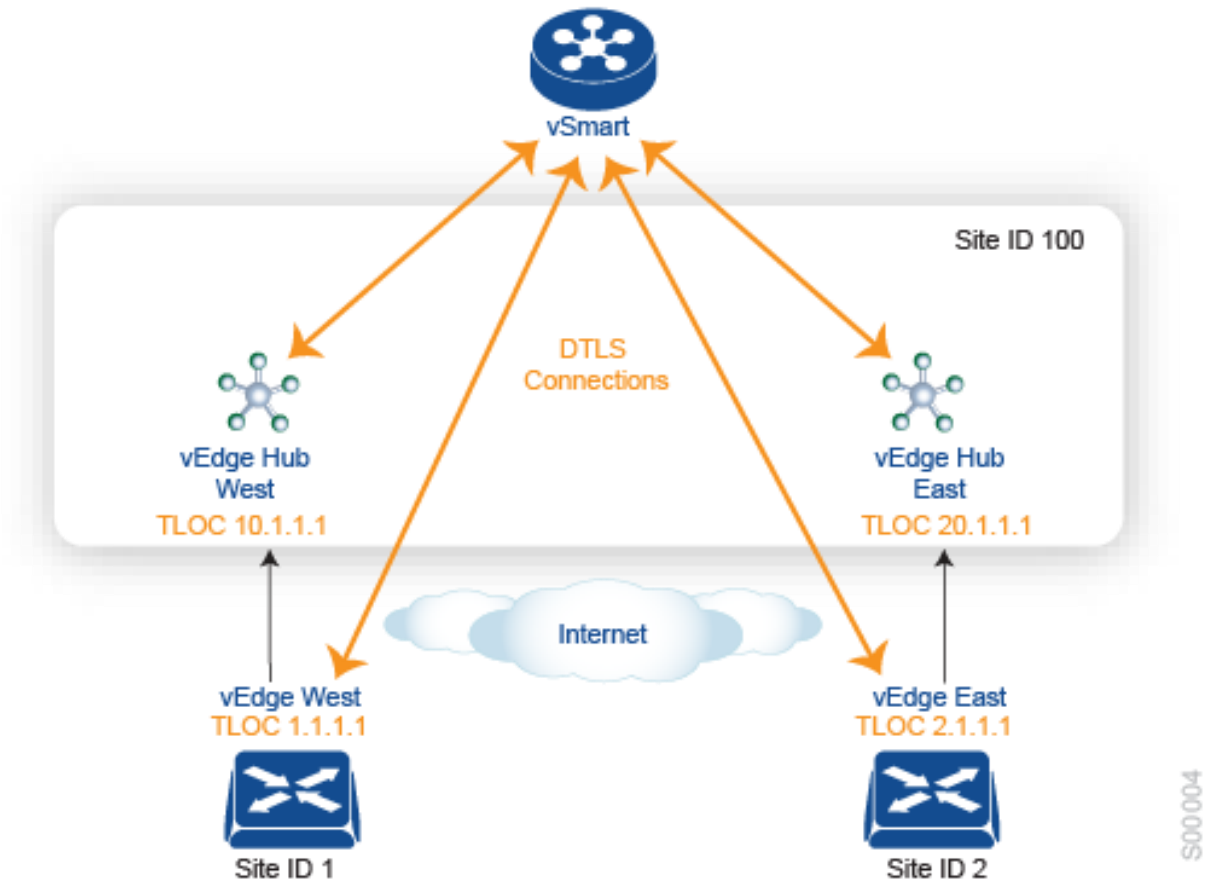
If the West side of the network had many sites instead of just one and each site had its own vEdge router, it would be straightforward to apply this same policy to all the sites. To do this, you simply add the site IDs of all the sites in the **site-list west-sites** list. This is the only change you need to make in the policy to have all the West side sites send traffic bound for the prefix 8.8.0.0/16 through the vEdge hub router. For example:

```
policy
  lists
    prefix-list east-prefixes
      ip-prefix 8.8.0.0/16
    site-list west-sites
      site-id 1
      site-id 11
      site-id 12
      site-id 13
  control-policy change-tloc
    sequence 10
      match route
        prefix-list east-prefixes
        site-id 2
      action accept
        set tloc 10.1.1.1 gold encap ipsec
apply-policy
  site west-sites control-policy change-tloc out
```

Creating Arbitrary Topologies

To provide redundancy in the hub-and-spoke-style topology discussed in the previous example, you can add a second vEdge hub to create a dual-homed hub site. The following figure shows that site ID 10 now has two vEdge hubs. We still want all inter-branch traffic to be routed through a vEdge hub. However, because we now have dual-homed hubs, we want to share the data traffic between the two hub routers.

- vEdge hub West, with TLOC 10.1.1.1, gold. We want all data traffic from branches on the West side of the overlay network to pass through and be processed by this vEdge router.
- vEdge hub East, with TLOC 20.1.1.1, gold. Similarly, we all East side data traffic to pass through the East vEdge hub.



Here is a policy configuration on the vSmart controller that would send West side data traffic through vEdge hub West and East side traffic through vEdge hub East:

```
policy
  lists
    site-list west-sites
      site-id 1
    site-list east-sites
      site-id 2
    tloc-list west-hub-tlocs
      tloc-id 10.1.1.1 gold
    tloc-list east-hub-tlocs
      tloc-id 20.1.1.1 gold
  control-policy prefer-west-hub
    sequence 10
      match tloc
        tloc-list west-hub-tlocs
      action accept
        set preference 50
  control-policy prefer-east-hub
    sequence 10
      match tloc
        tloc-list east-hub-tlocs
      action accept
        set preference 50
apply-policy
  site west-sites control-policy prefer-west-hub out
  site east-sites control-policy prefer-east-hub out
```

Here is an explanation of this policy configuration:

- Create the site lists that are required for the **apply-policy** configuration command:
  - **site-list west-sites** lists all the site IDs for all the vEdge routers in the West portion of the overlay network.
  - **site-list east-sites** lists the site IDs for the vEdge routers in the East portion of the network.
- Create the TLOC lists that are required for the match condition in the control policy:
  - **west-hub-tlocs** lists the TLOC for the West vEdge hub, which we want to service traffic from the West side vEdge routers.
  - **east-hub-tlocs** lists the TLOC for the East vEdge hub, to service traffic from the East vEdge routers.
- Define two control policies:
  - **prefer-west-hub** affects OMP routes destined to TLOC 10.1.1.1, gold, which is the TLOC address of the West vEdge hub router. This policy modifies the preference value in the OMP route to a value of 50, which is large enough that it is likely that no other OMP routes will have a larger preference. So setting a high preference value directs traffic destined for site 100 to the West hub router.
  - Similarly, **prefer-east-hub** sets the preference to 50 for OMP routes destined TLOC 20.1.1.1, gold, which is the TLOC address of the East vEdge hub router, thus directing traffic destined for site 100 site to the East vEdge hub router.
- Apply the control policies:
  - The first line in the **apply-policy** configuration has the vSmart controller apply the **prefer-west-hub** control policy to the sites listed in the **west-sites** list, which here is only site ID 1, so that the preference in their OMP routes destined to TLOC 10.1.1.1 is changed to 50 and traffic sent from the West vEdge routers to the hub site goes through the West vEdge hub router.
  - The vSmart controller applies the **prefer-east-hub** control policy to the OMP routes that it advertises to the vEdges in the **east-sites** list, which changes the preference to 50 for OMP routes destined to TLOC 20.1.1.1, so that traffic from the East vEdge routers goes to the East vEdge hub router.

Additional Information

Configuring Centralized Control Policy

Localized Control Policy

Control policy operates on the control plane traffic in the Viptela overlay network, influencing the determination of routing paths through the overlay network. Localized control policy is policy that is configured on a vEdge router (hence, it is local) and affects BGP and OSPF routing decisions on the site-local network that the vEdge router is part of.

In addition to participating in the overlay network, a vEdge router participates in the network at its local site, where it appears to the other network devices to be simply a regular router. As such, you can provision routing protocols, such as BGP and OSPF, on the vEdge router so that it can exchange route information with the local-site routers. To control and modify the routing behavior on the local network, you configure a type of control policy called route policy on the vEdge routers. Route policy applies only to routing performed at the local branch, and it affects only the route table entries in the local vEdge router's route table.

Additional Information

Configuring Localized Control Policy
Policy Overview

Configuring Localized Control Policy

This article provides procedures for configuring localized control policy. Localized control policy, configured on vEdge routers, lets you affect routing policy on the network at the local site where the vEdge router is located. This type of control policy is called route policy.

A route policy consists of a series of numbered (ordered) sequences of match-action pair that are evaluated in order, from lowest sequence number to highest sequence number. When a packet matches one of the match conditions, the associated action is taken and policy evaluation on that packets stops. Keep this in mind as you design your policies to ensure that the desired actions are taken on the items subject to policy.

If a packet matches no parameters in any of the sequences in the policy configured, it is, by default, rejected and discarded.

General vManage Configuration Procedure

To configure a localized control policy, also called a route policy, in vManage NMS, perform the following steps:

1. Create a localized policy.
2. Apply the localized policy in a device template.

Create a Localized Policy

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Localized Policy.
3. Click Add CLI Policy. The Add CLI Policy screen opens.
4. In the Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
5. In the Description field, enter a description for the route policy. It can contain up to 2048 characters. This field is mandatory, and it can contain any characters and spaces.
6. Enter the policy configuration in CLI format.
7. Click Create Variable to create a variable to use in the configuration.
8. Click Select a File to import a file containing policy configuration commands.
9. Click Add. The Configuration ► Policies screen opens, and the policies table includes the newly created policy.

Apply a Route Policy in a Device Template

1. n vManage NMS, select the Configuration ► Templates screen.
2. If you are creating a new device template:
    a. In the Device tab, click Create Template.
    b. From the Create Template drop-down, select From Feature Template.
    c. From the Device Model drop-down, select one of the vEdge devices.

d. In the Template Name field, enter a name for the device template. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.

e. In the Description field, enter a description for the device template. This field is mandatory, and it can contain any characters and spaces.

f. Continue with Step 4.

3. If you are editing an existing device template:

a. In the Device tab, click the More Actions icon to the right of the desired template, and click the pencil icon.

b. Click the Additional Templates tab. The screen scrolls to the Additional Templates section.

c. From the Policy drop-down, select the name of a policy that you have configured.

4. Click the Additional Templates tab located directly beneath the Description field. The screen scrolls to the Additional Templates section.

5. From the Policy drop-down, select the name of the policy you configured in the above procedure.

6. To apply a route policy to BGP:

a. Scroll to the Service VPN section.

b. In the Service VPN drop-down, type the service VPN number (a VPN number other than 0 or 512).

c. From Additional VPN Templates, select BGP.

d. From the BGP drop-down, click Create Template or View Template.

e. Select the Neighbor tab, click the plus sign (+), and click More.

f. In Address Family, change the scope to Device Specific. Then, Click On to enable Address Family, Click On to enable Route Policy In, and specify the name of a route policy to apply to prefixes received from the neighbor, or click On to enable Route Policy Out, and specify the name of a route policy to apply to prefixes sent to the neighbor. This name is one that you configured with a **policy route-policy** command.

g. Click Save to save the neighbor configuration, and then click Save to save the BGP configuration.

7. To apply a route policy to routes coming from all OSPF neighbors:

a. Scroll to the Service VPN section.

b. In the Service VPN drop-down, type the service VPN number (a VPN number other than 0 or 512).

c. From Additional VPN Templates, select OSPF.

d. Click Create Template or View Template.

e. Select the Advanced tab.

f. In Policy Name, specify the name of a route policy to apply to incoming routes. This name is one that you configured with a **policy route-policy** command.

g. Click Save.

8. To apply a route policy before redistributing routes into OSPF:

a. Scroll to the Service VPN section.

b. In the Service VPN drop-down, type the service VPN number (a VPN number other than 0 or 512).

c. From Additional VPN Templates, select OSPF.

d. Click Create Template or View Template.

e. Select the Redistribute tab, click the plus sign (+), and select the protocol from which to redistribute routes into OSPF.

f. Specify the name of a route policy to apply to the routes being redistributed. This name is one that you configured with a **policy route-policy** command.

g. Click Save.

9. Click Save (for a new template) or Update (for an existing template).

General CLI Configuration Procedure

To configure a route policy using the CLI:

1. Create lists of prefixes, as needed:
   vEdge(config)# **policy lists**
   vEdge(config-lists)# **prefix-list** *list-name*
   vEdge(config-lists- *list-name*)# **ip-prefix** *prefix* / *length*

2. Create lists of BGP AS paths, and community and extended community attributes, as needed:
   vEdge(config)# **policy lists**
   vEdge(config-lists)# **as-path-list** *list-name*
   vEdge(config-lists- *list-name*)# **as-path** *path-list*
   vEdge(config)# policy lists
   vEdge(config-lists)# **community-list** *list-name*
   vEdge(config-lists- *list-name*)# **community** [*aa* : *nn* | **internet** | **local-as** | **no-advertise** | **no-export**]
   vEdge(config-lists)# **ext-community-list** *list-name*
   vEdge(config-lists- *list-name*)# **community** [**rt** (*aa* : *nn* | *ip-address*) | **soo** (*aa* : *nn* | *ip-address*)]

3. Create a route policy instance:
   vEdge(config)# **policy route-policy** *policy-name*
   vEdge(config-route-policy- *policy-name*)#

4. Create a series of match–action pair sequences:
   vEdge(config-route-policy- *policy-name*)# **sequence** *number*
   vEdge(config-sequence- *number*)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

5. Define match parameters for routes:
   vEdge(config-sequence- *number*)# **match** *match- parameter*

6. Define actions to take when a match occurs:
   vEdge(config-sequence- *number*)# **action reject**
   vEdge(config-sequence- *number*)# **action accept set** *parameter*

7. Create additional numbered sequences of match–action pairs within the router policy, as needed.

8. If a route does not match any of the conditions in one of the sequences, it is rejected by default. To accept nonmatching routes, configure the default action for the policy:
   vEdge(config- *policy-name*)# **default-action accept**

9. Apply the policy to a BGP address family, to all OSPF inbound routes, or when redistributing OSPF routes:
   vEdge(config)# **vpn** *vpn-id* **router bgp** *local-as-number* **neighbor** *address*
   vEdge(config-neighbor)# **address-family ipv4-unicast**
   vEdge(config-address-family-ipv4-unicast)# **route-policy** *policy-name* (**in** | **out**)

   vEdge(config)# **vpn** *vpn-id* **router ospf**

vEdge(config-ospf)# **route-policy** *policy-name* **in**

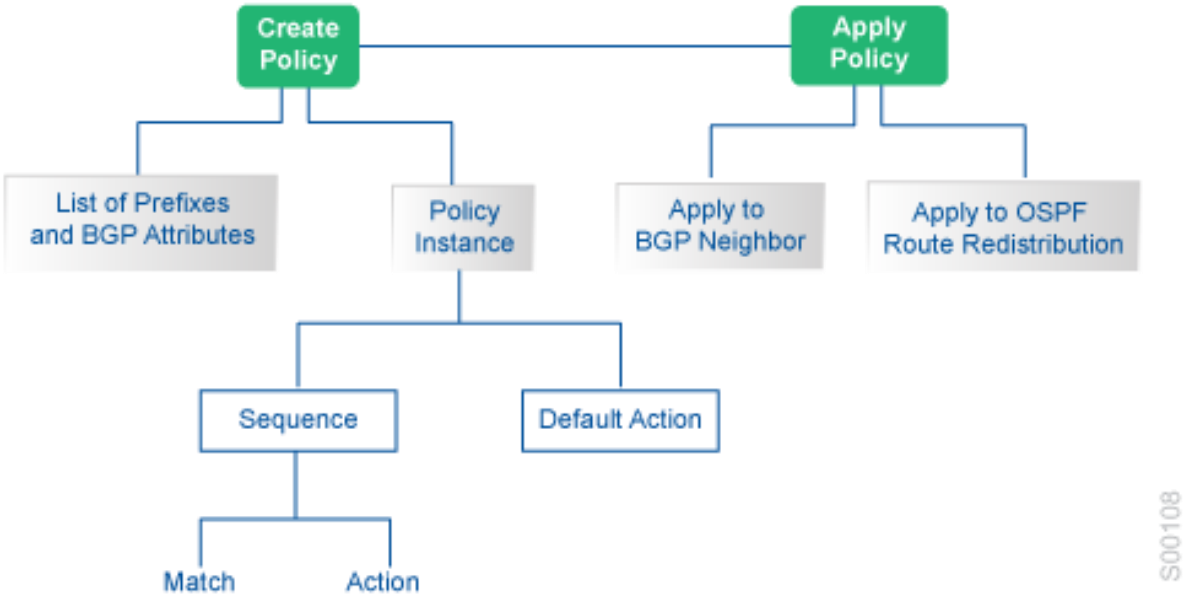
vEdge(config)# **vpn** *vpn-id* **router ospf**
vEdge(config-ospf)# **redistribute** (**bgp** | **connected** | **nat** | **omp** | **static**) **route-policy** *policy-name*

Structural Components of Policy Configuration for Localized Control Policy

Following are the structural components required to configure localized control policy. Each one is explained in more detail in the sections below.

```
policy
  lists
    as-path-list list-name
      as-path path-list
    community-list list-name
      community [aa:nn | internet | local-as | no-advertise | no-export]
    ext-community-list list-name
      community [rt (aa:nn | ip-address) | soo (aa:nn | ip-address)]
    prefix-list list-name
      ip-prefix prefix/length
  route-policy policy-name
    sequence number
      match
        match-parameters
      action
        reject
        accept
          set parameters
    default-action
      (accept | reject)
 vpn vpn-id router bgp local-as-number neighbor address
   address-family ipv4-unicast
     route-policy policy-name (in | out)
 vpn vpn-id router ospf
   route-policy policy-name in
   redistribute (bgp | connected | nat | omp | static) route-policy policy-name
```

The following figure illustrates the configuration components for localized control policy.



Lists

Route policy uses the following types of lists to group related items. You configure lists under the **policy lists** command hierarchy on vEdge routers.

List Type	Description	Command
AS paths	List of one or more BGP AS paths. You can write each AS as a single number or as a regular expression. To specify more than one AS in a single path, include the list in quotation marks (" "). To configure multiple AS paths in a single list, include multiple **as-path** options, specifying one AS path in each option.	**as-path-list** *list-name* **as-path** *path-list*
Communities	List of one of more BGP communities. In **community** , you can specify: • *aa* : *nn* : Autonomous system number and network number. Each number is a 2-byte value with a range from 1 to 65535. • **internet** : Routes in this community are advertised to the Internet community. This community comprises all BGP-speaking networking devices. • **local-as** : Routes in this community are not advertised outside the local AS. • **no-advertise** : Attach the NO_ADVERTISE community to routes. Routes in this community are not advertised to other BGP peers. • **no-export** : Attach the NO_EXPORT community to routes. Routes in this community are not advertised outside the local AS or outside a BGP confederation boundary. To configure multiple BGP communities in a single list, include multiple **community** options, specifying one community in each option.	**community-list** *list-name* **community** [*aa* : *nn* | **internet** | **local-as** | **no-advertise** | **no-export**]
Extended communities	List of one or more BGP extended communities. In **community** , you can specify: • **rt** (*aa* : *nn* | *ip-address*): Route target community, which is one or more routers that can receive a set of routes carried by BGP. Specify this as the autonomous system number and network number, where each number is a 2-byte value with a range from 1 to 65535, or as an IP address. • **soo** (*aa* : *nn* | *ip-address*): Route origin community, which is one or more routers that can inject a set of routes into BGP. Specify this as the autonomous system number and network number, where each number is a 2-byte value with a range from 1 to 65535, or as an IP address. To configure multiple extended BGP communities in a single list, include multiple **community** options, specifying one community in each option.	**ext-community-list** *list-name* **community** [**rt** (*aa* : *nn* | *ip-address*) | **soo** (*aa* : *nn* | *ip-address*)]

Prefixes	List of one or more IP prefixes. To configure multiple prefixes in a single list, include multiple **ip-prefix** options, specifying one prefix in each option. Specify the IP prefixes as follows: • *prefix* / *length* —Exactly match a single prefix–length pair. • **0.0.0.0/0** —Match any prefix–length pair. • **0.0.0.0/0 le** *length* —Match any IP prefix whose length is less than or equal to *length* . For example, **ip-prefix 0.0.0.0/0 le 16** matches all IP prefixes with lengths from /1 through /16. • **0.0.0.0/0 ge** *length* —Match any IP prefix whose length is greater than or equal to *length* . For example, **ip-prefix 0.0.0.0 ge 25** matches all IP prefixes with lengths from /25 through /32. • **0.0.0.0/0 ge** *length1* **le** *length2* , or **0.0.0.0 le** *length2* **ge** *length1* —Match any IP prefix whose length is greater than or equal to *length1* and less than or equal to *length2* . For example, **ip-prefix 0.0.0.0/0 ge 20 le 24** matches all /20, /21, /22, /23, and /24 prefixes. Also, **ip-prefix 0.0.0.0/0 le 24 ge 20** matches the same prefixes. If *length1* and *length2* are the same, a single IP prefix length is matched. For example, **ip-prefix 0.0.0.0/0 ge 24 le 24** matches only /24 prefixes.	**prefix-list** *list-name* **ip-prefix** *prefix* / *length*

Sequences

A localized control policy contains sequences of match–action pairs. The sequences are numbered to set the order in which a route is analyzed by the match–action pairs in the policy. You configure sequences with the **route-policy** sequence command.

Each sequence in a localized control policy can contain one **match** command and one **action**

Match Parameters

For route policy routes, you can configure these parameters under the **match** command:

Description	Command	Value or Range
IP prefix or prefixes from which the route was learned	**address** *list-name*	Name of an IP prefix list
BGP AS paths	**as-path** *list-name*	Name of an AS path list
BGP communities	**community** *list-name*	Name of a BGP community list
BGP extended communities	**ext-community** *list-name*	Name of a BGP extended community list
Route metric	**metric** *number*	0 through 4294967295
Next hop	**next-hop** *list-name*	Name of an IP prefix list
OMP tag for OSPF	**omp-tag** *number*	0 through 4294967295
BGP origin code	**origin** *origin*	**egp** (default), **igp** , **incomplete**
OSPF tag value	**ospf-tag** *number*	0 through 4294967295
Peer address	**peer** *address*	IP address

Action Parameters

When a route matches the conditions in the match portion of a route policy, the route can be accepted or rejected:

Description	Command	Value or Range

Accept the route. An accepted route is eligible to be modified by the additional parameters configured in the **action** portion of the policy configuration.	**accept**	—
Discard the packet.	**reject**	—

For a route that is accepted, the following actions can be configured:

Description	Parameter	Value or Range
Set the AS number in which a BGP route aggregator is located and the IP address of the route aggregator.	**set aggregator** *as-number ip-address*	0 through 65535
Set an AS number or a series of AS numbers to exclude from the AS path or to prepend to the AS path.	**set as-path** (**exclude** | **prepend**) *as-number*	0 through 65535
Set the BGP atomic aggregate attribute.	**set atomic-aggregate**	—
Set the BGP community value.	**set community** *value*	[*aa* : *nn* | **internet** | **local-as** | **no-advertise** | **no-export**]
Set the BGP local preference.	**set local-preference** *number*	0 through 4294967295
Set the metric value.	**set metric** *number*	0 through 4294967295
Set the metric type.	**set metric-type** *type*	**type1** , **type2**
Set the next-hop address.	**set next-hop** *ip-address*	IP address
Set the OMP tag for OSPF to use.	**set omp-tag** *number*	0 through 4294967295
Set the BGP origin code.	**set origin** *origin*	**egp** , **igp** (default), **incomplete**
Set the IP address from which the route was learned.	**set originator** *ip-address*	IP address
Set the OSPF tag value.	**set ospf-tag** *number*	0 through 4294967295
Set the BGP weight.	**set weight** *number*	0 through 4294967295

To display the OMP and OSPF tag values associated with a route, use the **show ip routes detail** command.

Defining the Default Action

If a route being evaluated does not match any of the match conditions in a control policy, a default action is applied to this route. By default, the route is rejected. To modify this behavior, include the **default-action accept** command in the control policy.

Applying Route Policy for BGP

For a route policy to take effect for BGP, you must apply it to an address family. Currently, the Viptela software supports only the IPv4 address family, so you apply route policy with this command:

```
vEdge(config)# vpn vpn-id router bgp local-as-number neighbor address address-family ipv4-unicast route-policy policy-name (in | out)
```

Applying the policy in the inbound direction (**in**) affects routes being received by BGP. Applying the policy in the outbound direction (**out**) affects routes being advertised by BGP.

Applying Route Policy for OSPF

For a route policy to take effect for OSPF, you can apply it to all inbound traffic:

```
vEdge(config)# vpn vpn-id router ospf route-policy policy-name in
```

You can also apply the policy when redistributing routes into OSPF:

```
vEdge(config)# vpn vpn-id router ospf redistribute (bgp | connected | nat | omp | static) route-policy
policy-name
```

Additional Information

Localized Control Policy

Centralized Data Policy

Centralized data policy is policy that is configured on a vSmart controller (hence, it is centralized) and that affects data traffic being transmitted between the routers on the Viptela overlay network.

Centralized Data Policy Overview

Data policy operates on the data plane in the Viptela overlay network and affects how data traffic is sent among the vEdge routers in the network. The Viptela architecture defines two types of data policy, centralized data policy, which controls the flow of data traffic based on the IP header fields in the data packets and based on network segmentation, and localized data policy, which controls the flow of data traffic into and out of interfaces and interface queues on a vEdge router.

Centralized data policy is applied to packets that originate from a specific sender, or source address, for instance, from a workstation in a local site that is sending voice, data, or other traffic, and it controls which destinations within a VPN the traffic can reach. Data policy is applied to data traffic based on a 6-tuple of fields in the packet's IP header: source IP address, source port, destination IP address, destination port, DSCP, and protocol.

As with control policy, data policy is provisioned centrally on a vSmart controller and is applied only on the vSmart controller. The data policy itself is never pushed to the vEdge routers in the network. What is pushed to the vEdge devices, via OMP and based on the site ID, are the results of the data policy; hence, the effects of the policy are reflected on the vEdge routers. Normally, the data policy on a vEdge router acts as the data policy for the entire site that sits behind the router. Data policy that comes from the vSmart controller is always implicitly applied in the inbound direction.

Data policy can be applied to data traffic based on the packet header fields, such as the prefix, port, protocol, and DSCP value, and they can also be applied based on the VPN in the overlay network to which the traffic belows.

Data Policy Based on Packet Header Fields

Policy decisions affecting data traffic can be based on the packet header fields, specifically, on the source and destination IP prefixes, the source and destination IP ports, the protocol, and the DSCP.

This type of policy is often used to modify traffic flow in the network. Here are some examples of the types of control that can be effected with centralized data policy:

- Which set of sources are allowed to send traffic to any destination outside the local site. For example, local sources that are rejected by such a data policy can communicate only with hosts on the local network.
- Which set of sources are allowed to send traffic to a specific set of destinations outside the local site. For example, local sources that match this type of data policy can send voice traffic over one path and data traffic over another.
- Which source addresses and source ports are allowed to send traffic to any destination outside the local site or to a specific port at a specific destination.

VPN Membership Policy

A second type of centralized data policy is VPN membership policy. It controls whether a vEdge router can participate in a particular VPN. Stated another way, VPN membership policy defines which VPNs a vEdge router is and is not allowed to receive routes from.

VPN membership policy can be centralized, because it affects only the packet headers and has no impact on the choice of interface that a vEdge router uses to transmit traffic. What happens instead is that if, because of a VPN membership policy, a vEdge router is not allowed to receive routes for a particular VPN, the vSmart controller never forwards those routes to that router.

Deep Packet Inspection

In addition to examining the network- and transport-layer headers in data packets, centralized data policy can be used to examine the application information in the data packets' payload. This deep packet inspection offers control over how data packets from specific applications or application families are forwarded across the network, allowing you to assign the traffic to be carried by specific tunnels. To control the traffic flow of specific application traffic based on the traffic loss or latency properties on a tunnel, use application-aware routing.

Additional Information

Application-Aware Routing
Centralized Data Policy Configuration Examples
Configuring Centralized Data Policy
Configuring Split DNS
Policy Overview
Traffic Flow Monitoring with Cflowd
Using a vEdge Router as a NAT Device

Configuring Centralized Data Policy

This article provides general procedures for configuring centralized data policy on vSmart controllers. Centralized data policy can be used for different purposes, which are described in separate sections in this article:

- To base policy decisions on source and destination prefixes and on the headers in the IP data packets, you use centralized data policy, which you configure with the **policy data-policy** command. The vSmart controller pushes this type of data policy to the vEdge routers. In domains with multiple vSmart controllers, all the controllers must have the same centralized data policy configuration to ensure that traffic flow within the overlay network remains synchronized.
- To base policy decisions on the application information in the packet payload, you use centralized data policy to perform deep packet inspection. You configure this by creating lists of applications with the **policy lists app-list** command and then calling these lists in a **policy data-policy** command. To specify the path that application traffic takes through the network, you can set the local TLOC or the remote TLOC, or both, to use to send the traffic over.
- To configure the VPNs that vEdge routers are allowed to receive routes from, you use centralized data policy, which you configure with the **policy vpn-membership** command. VPN membership policy affects which routes the vSmart controller sends to the vEdge router. The policy itself remains on the vSmart controller and is not pushed to the vEdge routers.

Configuring Centralized Data Policy Based on Prefixes and IP Headers

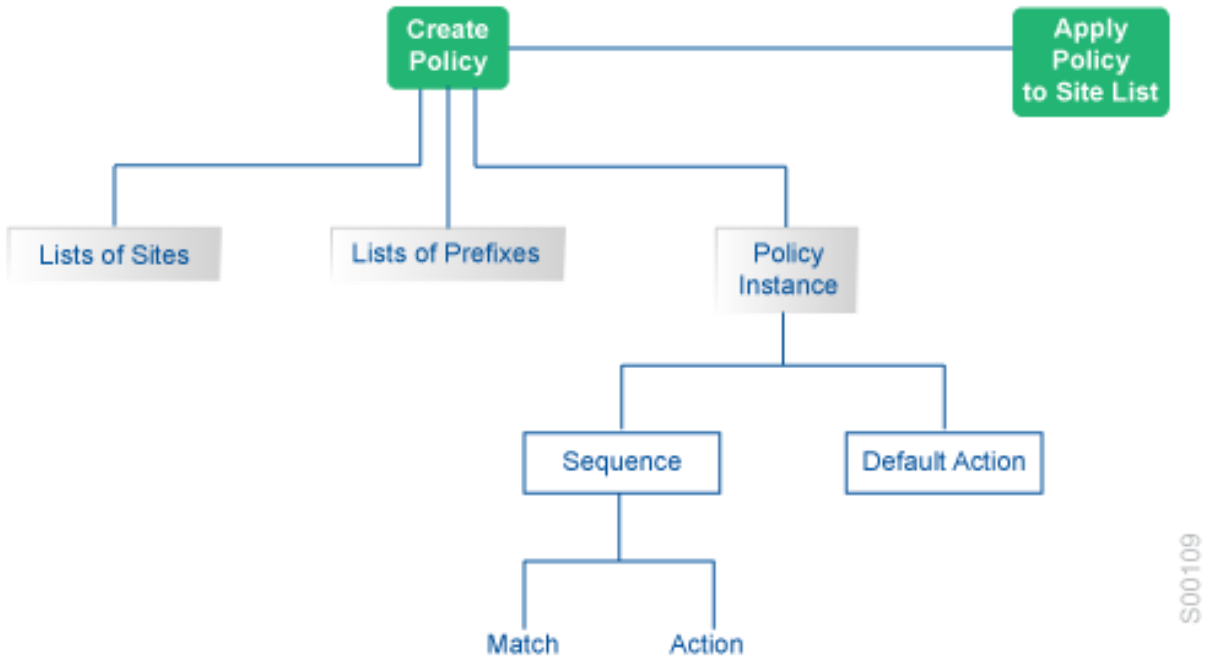
A centralized data policy based on source and destination prefixes and on headers in IP packets consists of a series of numbered (ordered) sequences of match-action pair that are evaluated in order, from lowest sequence number to highest sequence number. When a packet matches one of the match conditions, the associated action is taken and policy evaluation on that packets stops. Keep this in mind as you design your policies to ensure that the desired actions are taken on the items subject to policy.

If a packet matches no parameters in any of the sequences in the policy configuration, it is dropped and discarded by default.

To create a centralized data policy to filter based on IP prefixes and IP packet headers, you include the following components in the configuration on a vSmart controller:

Component	Description	Configuration Command
Lists	Groupings of related items that you reference in the match and action portions of the data policy configuration. For centralized data policy, you can group applications, IP prefixes, sites, TLOCs, and VPNs.	**policy lists**
Centralized data policy instance	Container for centralized data policy that f ilters packets based on IP prefix and IP packet header fields.	**policy data-policy**
VPN list	List of VPNs to which to apply the centralized data policy.	**policy data-policy vpn-list**
Numbered sequences of match–action pairs	Sequences that establish the order in which the policy components are applied	**policy data-policy vpn-list sequence**
Match parameters	Conditions that packets must match to be considered for a data policy.	**policy data-policy vpn-list sequence match**
Actions	Whether to accept or reject (and drop) matching packets, and how to process matching packets.	**policy data-policy vpn-list sequence action**
Default action	Action to take if a packet matches none of the policy conditions.	**policy data-policy vpn-list default-action**
Application of centralized data policy	For a data policy to take effect, you apply it to one or more sites in the overlay network.	**apply-policy site-list data-policy**

The following figure illustrates the configuration components for centralized data policy:



General vManage Configuration Procedure

To configure centralized data policies, use the vManage policy configuration wizard. The wizard consists of four sequential screens that guide you through the process of creating and editing policy components:

1. Create Applications or Groups of Interest—Create lists that group together related items and that you call in the match or action components of a policy.
2. Configure Topology—Create the network structure to which the policy applies.
3. Configure Traffic Rules—Create the match and action conditions of a policy.
4. Apply Policies to Sites and VPNs—Associate policy with sites and VPNs in the overlay network.

In the first three policy configuration wizard screens, you are creating policy components or blocks. In the last screen, you are applying policy blocks to sites and VPNs in the overlay network.

For a centralized data policy to take effect, you must activate the policy.

Start the Policy Configuration Wizard

To start the policy configuration wizard:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy.

The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed.

Create Applications or Groups of Interest

To create lists of applications or groups to use in centralized data policy:

1. Start the policy configuration wizard as explained above.
2. Create new lists , as described in the following table:

List Type	Procedure
Application	1. In the left bar, click Application. 2. Click New Application List. 3. Enter a name for the list. 4. Click either the Application or Application Family button. 5. From the Select drop-down, select the desired applications or application families. 6. Click Add. Two application lists are preconfigured. You cannot edit or delete these lists. 7. Google_Apps—Includes Google applications, such as gmail, Google maps, and YouTube. To display a full list of Google applications, click the list in the Entries column. 8. Microsoft_Apps—Includes Microsoft applications, such as Excel, Skype, and Xbox. To display a full list of Microsoft applications, click the list in the Entries column.

Data Prefix	9. In the left bar, click Data Prefix. 10. Click New Data Prefix List. 11. Enter a name for the list. 12. In the Add Data Prefix field, enter one or more data prefixes separated by commas. 13. Click Add.
Policer	14. In the left bar, click Policer. 15. Click New Policer List. 16. Enter a name for the list. 17. Define the policing parameters: a. In the Burst field, enter the maximum traffic burst size, a value from 15,000 to 10,000,000 bytes. b. In the Exceed field, select the action to take when the burst size or traffic rate is exceeded. It can be drop, which sets the packet loss priority (PLP) to low, or remark, which sets the PLP to high. c. In the Rate field, enter the maximum traffic rate, a value from 0 through $2^{64} - 1$ bits per second (bps). 18. Click Add.
Prefix	19. In the left bar, click Prefix. 20. Click New Prefix List. 21. Enter a name for the list. 22. In the Add Prefix field, enter one or more data prefixes separated by commas. 23. Click Add.
Site	24. In the left bar, click Site. 25. Click New Site List. 26. Enter a name for the list. 27. In the Add Site field, enter one or more site IDs separated by commas. 28. Click Add.

TLOC	29. In the left bar, click TLOC. 30. Click New TLOC List. The TLOC List popup displays. 31. Enter a name for the list. 32. In the TLOC IP field, enter the system IP address for the TLOC. 33. In the Color field, select the TLOC's color. 34. In the Encap field, select the encapsulation type. 35. In the Preference field, optionally select a preference to associate with the TLOC. 36. Click Add TLOC to add another TLOC to the list. 37. Click Save.
VPN	38. In the left bar, click VPN. 39. Click New VPN List. 40. Enter a name for the list. 41. In the Add VPN field, enter one or more VPN IDs separated by commas. 42. Click Add.

4. Click Next to move to Configure Topology in the wizard. When you first open this screen, the Topology tab is selected by default.

Configure the Network Topology

To configure the network topology or a VPN membership to use in centralized data policy:

1. If you are already in the policy configuration wizard, skip to Step 4. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Create a network topology, as described in the following table:

Policy Type	Description	Procedure

Hub and Spoke	Policy for a topology with one or more central hub sites and with spokes connected to a hub	1. In the Add Topology drop-down, select Hub and Spoke. 2. Enter a name for the hub-and-spoke policy. 3. Enter a description for the policy. 4. In the VPN List field, select the VPN list for the policy. 5. In the left pane, click Add Hub and Spoke. A hub-and-spoke policy component containing the text string My Hub-and-Spoke is added in the left pane. 6. Double-click the My Hub-and-Spoke text string, and enter a name for the policy component. 7. In the right pane, add hub sites to the network topology: a. Click Add Hub Sites. b. In the Site List Field, select a site list for the policy component. c. Click Add. d. Repeat Steps 7a, 7b, and 7c to add more hub sites to the policy component. 8. In the right pane, add spoke sites to the network topology: a. Click Add Spoke Sites. b. In the Site List Field, select a site list for the policy component. c. Click Add. d. Repeat Steps 8a, 8b, and 8c to add more spoke sites to the policy component. 9. Repeat Steps 5 through 8 to add more components to the hub-and-spoke policy. 10. Click Save Hub and Spoke Policy.
Mesh	Partial-mesh or full-mesh region	11. In the Add Topology drop-down, select Mesh. 12. Enter a name for the mesh region policy component. 13. Enter a description for the mesh region policy component. 14. In the VPN List field, select the VPN list for the policy. 15. Click New Mesh Region. 16. In the Mesh Region Name field, enter a name for the individual mesh region. 17. In the Site List field, select one or more sites to include in the mesh region. 18. Repeat Steps 5 through 7 to add more mesh regions to the policy. 19. Click Save Mesh Region.

5. To use an existing topology:

a. In the Add Topology drop-down, click Import Existing Topology. The Import Existing Topology popup displays.

b. Select the type of topology.

c. In the Policy drop-down, select the name of the topology.

d. Click Import.

6. To create a VPN membership policy, in the Topology bar, click VPN Membership. Then:

a. Click Add VPN Membership Policy. The Update VPN Membership Policy popup displays.

b. Enter a name and description for the VPN membership policy.

c. In the Site List field, select the site list.

d. In the VPN Lists field, select the VPN list.

e. Click Add List to add another VPN to the VPN membership.

f. Click Save

7. Click Next to move to Configure Traffic Rules in the wizard. When you first open this screen, the Application-Aware Routing tab is selected by default.

Configure Traffic Rules

To create the match and action rules to apply to traffic affected by the policy:

1. If you are already in the policy configuration wizard, skip this procedure. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.

To configure traffic rules for centralized data policy:

1. In the Application-Aware Routing bar, select the Traffic Data tab.
2. Click the Add Policy drop-down.
3. Select Create New. The Add Data Policy popup opens.
4. Select the type of data policy from Application Firewall, QoS, Service Chaining, Traffic Engineering, and Custom.
5. In the left pane, click Sequence Type. A policy sequence containing the text string Application Firewall, QoS, Service Chaining, Traffic Engineering, or Custom is added in the left pane.
6. Double-click the text string, and enter a name for the policy sequence.The name you type is displayed both in the Sequence Type list in the left pane and in the right pane.
7. In the right pane, click Sequence Rule. The Match/Action box opens, and Match is selected by default. The available policy match conditions are listed below the box.
8. To select one or more Match conditions, click its box and set the values as described in the following table. Note that not all match conditions are available for all policy sequence types.

Match Condition	Procedure

None (match all packets)	Do not specify any match conditions.
Applications/Application Family List	1. In the Match conditions, click Applications/Application Family List. 2. In the drop-down, select the application family. 3. To create an application list: a. Click New Application List. b. Enter a name for the list. c. Click the Application button to create a list of individual applications. Click the Application Family to create a list of related applications. d. In the Select Application drop-down, select the desired applications or application families. e. Click Save.
Destination Data Prefix	4. In the Match conditions, click Destination Data Prefix. 5. To match a list of destination prefixes, select the list from the drop-down. 6. To match an individual destination prefix, type the prefix in the Destination box.
Destination Port	7. In the Match conditions, click Destination Port. 8. In the Destination field, enter the port number. Specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-]).
DNS Application List (to enable split DNS)	9. In the Match conditions, click DNS Application List. 10. In the drop-down, select the application family.
DNS (to enable split DNS)	11. In the Match conditions, click DNS. 12. In the drop-down, select Request to process DNS requests for the DNS applications, and select Response to process DNS responses for the applications.
DSCP	13. In the Match conditions, click DSCP. 14. In the DSCP field, type the DSCP value, a number from 0 through 63.
Packet Length	15. In the Match conditions, click Packet Length. 16. In the Packet Length field, type the length, a value from 0 through 65535.
PLP	17. In the Match conditions, click PLP. 18. In the PLP drop-down, select Low or High. To set the PLP to high, apply a policer that includes the **exceed remark** option.
Protocol	19. In the Match conditions, click Protocol. 20. In the Protocol field, type the Internet Protocol number, a number from 0 through 255.
Source Data Prefix	21. In the Match conditions, click Source Data Prefix. 22. To match a list of source prefixes, select the list from the drop-down. 23. To match an individual source prefix, type the prefix in the Source box.

Source Port	24. In the Match conditions, click Source Port. 25. In the Source field, enter the port number. Specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-]).
TCP	26. In the Match conditions, click TCP. 27. In the TCP field, syn is the only option available.

9. To select actions to take on matching data traffic, click the Actions box.
10. To drop matching traffic, click the Drop button.
11. To accept matching traffic, click the Accept button. The available policy actions are listed to the right of the button.
12. Set the policy action as described in the following table:

Match Condition	Description	Procedure
Counter	Count matching data packets.	1. In the Action conditions, click Counter. 2. In the Counter Name field, enter the name of the file in which to store packet counters.
DSCP	Assign a DSCP value to matching data packets.	3. In the Action conditions, click DSCP. 4. In the DSCP field, type the DSCP value, a number from 0 through 63.
Forwarding Class	Assign a forwarding class to matching data packets.	5. In the Match conditions, click Forwarding Class. 6. In the Forwarding Class field, type the class value, which can be up to 32 characters long.
Log	Place a sampled set of packets that match the SLA class rule into system logging (syslog) files. In addition to logging the packet headers, a syslog message is generated the first time a packet header is logged and then every 5 minutes thereafter, as long as the flow is active.	7. In the Action conditions, click Log to enable logging.
Policer	Apply a policer to matching data packets.	8. In the Match conditions, click Policer. 9. In the Policer drop-down field, select the name of a policer.

13. Click Save Match and Actions.
14. Create additional sequence rules as desired. Drag and drop to re-arrange them.
15. Create additional sequence types as desired. Drag and drop to re-arrange them.
16. Click Save Data Policy.

Click Next to move to Apply Policies to Sites and VPNs in the wizard.

Apply Policies to Sites and VPNs

In the last screen of the policy configuration wizard, you associate the policy blocks that you created on the previous three screens with VPNs and with sites in the overlay network.

To apply a policy block to sites and VPNs in the overlay network:

1. If you are already in the policy configuration wizard, skip to Step 6. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.
5. Click Next. The Apply Policies to Sites and VPNs screen opens.
6. In the Policy Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
7. In the Policy Description field, enter a description of the policy. It can contain up to 2048 characters. This field is mandatory, and it can contain any characters and spaces.
8. From the Topology bar, select the type of policy block. The table then lists policies that you have created for that type of policy block.
9. Associate the policy with VPNs and sites. The choice of VPNs and sites depends on the type of policy block:
    a. For a Topology policy block, click Add New Site List and VPN List or Add New Site. Some topology blocks might have no Add buttons. Select one or more site lists, and select one or more VPN lists. Click Add.
    b. For an Application-Aware Routing policy block, click Add New Site List and VPN list. Select one or more site lists, and select one or more VPN lists. Click Add.
    c. For a Traffic Data policy block, click Add New Site List and VPN List. Select the direction for applying the policy (From Tunnel, From Service, or All), select one or more site lists, and select one or more VPN lists. Click Add.
    d. For a cflowd policy block, click Add New Site List. Select one or more site lists, Click Add.
10. Click Preview to view the configured policy. The policy is displayed in CLI format.
11. Click Save Policy. The Configuration ► Policies screen opens, and the policies table includes the newly created policy.

Activate a Centralized Data Policy

Activating a centralized data policy sends that policy to all connected vSmart controllers. To activate a centralized policy:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Select a policy.
3. Click the More Actions icon to the right of the row, and click Activate. The Activate Policy popup opens. It lists the IP addresses of the reachable vSmart controllers to which the policy is to be applied.
4. Click Activate.

General CLI Configuration Procedure

Following are the high-level steps for configuring a centralized data policy based on prefixes and the headers in the IP packets. By default, matching is done on the 6-tuple consisting of the source IP address, destination IP address, source port, destination port, protocol, and DSCP.

1. Create a list of overlay network sites to which the centralized data policy is to be applied (in the **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists- *list-name*)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).
   Create additional site lists, as needed.

2. Create lists of IP prefixes and VPNs, as needed:
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **data-prefix-list** *list-name*
   vSmart(config-lists- *list-name*)# **ip-prefix** *prefix* / *length*

   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists- *list-name*)# **vpn** *vpn-id*

3. Create lists of TLOCs, as needed:
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists tloc-list** *list-name*
   vSmart(config-lists- *list-name*)# **tloc** *ip-address* **color** *color* **encap** *encapsulation* [**preference** *number*]

4. Define policing parameters, as needed:
   vSmart(config-policy)# **policer** *policer-name*
   vSmart(config-policer)# **rate** *bandwidth*
   vSmart(config-policer)# **burst** *bytes*
   vSmart(config-policer)# **exceed** *action*

5. Create a data policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy data-policy** *policy-name*
   vSmart(config-data-policy- *policy-name*)# **vpn-list** *list-name*

6. Create a series of match–pair sequences:
   vSmart(config-vpn-list)# **sequence** *number*
   vSmart(config-sequence- *number*)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

7. Define match parameters for packets:
   vSmart(config-sequence- *number*)# **match** *parameters*

8. Define actions to take when a match occurs:
   vSmart(config-sequence- *number*)# **action** (**accept** | **drop**) [**count** *counter-name*] [**log**] [**tcp-optimization**]
   vSmart(config-sequence- *number*)# **action acccept nat** [**pool** *number*] [**use-vpn 0**]
   vSmart(config-sequence- *number*)# **action accept redirect-dns** (**host** | *ip-address*)
   vSmart(config-sequence- *number*)# **action accept set** *parameters*

9. Create additional numbered sequences of match–action pairs within the data policy, as needed.

10. If a route does not match any of the conditions in one of the sequences, it is rejected by default. To accept nonmatching prefixed, configure the default action for the policy:
    vSmart(config- *policy-name*)# **default-action accept**

11. Apply the policy to one or more sites in the overlay network:
    vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* (**all** | **from-service** | **from-tunnel**)

Structural Components of Policy Configuration for Centralized Data Policy

Following are the structural components required to configure centralized data policy based on IP addresses and prefixes. Each one is explained in more detail in the sections below.

```
policy
  lists
    app-list list-name
      (app applications | app-family application-families)
    data-prefix-list list-name
      ip-prefix prefix
    site-list list-name
      site-id site-id
    tloc-list list-name
      tloc ip-address color color encap encapsulation [preference value]
    vpn-list list-name
      vpn vpn-id
  policer policer-name
    burst bytes
    exceed action
    rate bandwidth
  data-policy policy-name
    vpn-list list-name
      sequence number
        match
          app-list list-name
          destination-data-prefix-list list-name
          destination-ip prefix/length
          destination-port port-numbers
          dscp number
          dns-app-list list-name
          dns (request | response)
          packet-length number
          protocol number
          source-data-prefix-list list-name
          source-ip prefix/length
          source-port port-numbers
          tcp flag
        action
          cflowd (not available for deep packet inspection)
          count counter-name
          drop
          log
          redirect-dns (dns-ip-address | host)
          tcp-optimization
          accept
            nat [pool number] [use-vpn 0]
            set
              dscp number
              forwarding-class class
              local-tloc color color [encap encapsulation] [restrict]
              next-hop ip-address
              policer policer-name
              service service-name local [restrict] [vpn vpn-id]
```

```
                service service-name [tloc ip-address | tloc-list list-name] [vpn vpn-id]
                tloc ip-address color color [encap encapsulation]
                tloc-list list-name
                vpn vpn-id
        default-action
          (accept | drop)
apply-policy site-list list-name
  data-policy policy-name (all | from-service | from-tunnel)
```

Lists

Centralized data policy uses the following types of lists to group related items. You configure lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command
Applications and application families	List of one or more applications or application families running on the subnets connected to the vEdge router. Each **app-list** can contain either applications or application families, but you cannot mix the two. To configure multiple applications or application families in a single list, include multiple **app** or **app-family** options, specifying one application or application family in each **app** or **app-family** option. • *application-name* is the name of an application. The Viptela software supports about 2300 different applications. To list the supported applications, use the **?** in the CLI. • *application-family* is the name of an application family. It can one of the following: **antivirus** , **application-service** , **audio_video** , **authentication** , **behavioral** , **compression** , **database** , **encrypted** , **erp** , **file-server** , **file-transfer** , **forum** , **game** , **instant-messaging** , **mail** , **microsoft-office** , **middleware** , **network-management** , **network-service** , **peer-to-peer** , **printer** , **routing** , **security-service** , **standard** , **telephony** , **terminal** , **thin-client** , **tunneling** , **wap** , **web** , and **webmail** .	**app-list** *list-name* (**app** *applications* | **app-family** *application-families*)
Data prefixes	List of one or more IP prefixes. To configure multiple prefixes in a single list, include multiple **ip-prefix** options, specifying one prefix in each option.	**data-prefix-list** *list-name* **ip-prefix** *prefix* / *length*
Sites	List of one or more site identifiers in the overlay network. To configure multiple sites in a single list, include multiple **site-id** options, specifying one site number in each option. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*
TLOCs	List of one or more address of transport locations (TLOCs) in the overlay network. For each TLOC, specify its address, color, and encapsulation. *address* is the system IP address. For each TLOC, specify its address, color, and encapsulation. *address* is the system IP address. *color* can be one of **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . *encapsulation* can be **gre** or **ipsec** . Optionally, set a preference value (from 0 to $2^{32} - 1$) to associate with the TLOC address. When you apply a TLOC list in an **action accept** condition, when multiple TLOCs are available and satisfy the match conditions, the TLOC with the lowest preference value is used. If two or more of TLOCs have the lowest preference value, traffic is sent among them in an ECMP fashion.	**tloc-list** *list-name* **tloc** *ip-address* **color** *color* **encap** *encapsulation* [**preference** *value*]
VPNs	List of one or more VPNs in the overlay network. For data policy, you can configure any VPNs except for VPN 0 and VPN 512. To configure multiple VPNs in a single list, include multiple **vpn** options, specifying one VPN number in each option. You can specify a single VPN identifier (such as **vpn 1**) or a range of VPN identifiers (such as **vpn 1-10**).	**vpn-list** *list-name* **vpn** *vpn-id*

In the vSmart controller configuration, you can create multiple iterations of each type of list. For example, it is common to create multiple site lists and multiple VPN lists so that you can apply data policy to different sites and different customer VPNs across the network.

You cannot apply the same type of policy to site lists that contain overlapping site IDs. That is, all data policies cannot have overlapping site lists among themselves. If you accidentally misconfigure overlapping site lists, the attempt to commit the configuration on the vSmart controller fails.

VPN Lists

Each centralized data policy is associated with a VPN list. You configure VPN lists with the **policy data-policy vpn-list** command. The list you specify must be one that you created with a **policy lists vpn-list** command.

For centralized data policy, you can include any VPNs except for VPN 0 and VPN 512. VPN 0 is reserved for control traffic, so never carries any data traffic, and VPN 512 is reserved for out-of-band network management, so also never carries any data traffic. Note that while the CLI allows you to include these two VPNs in a data policy configuration, the policy is not applied to these two VPNs.

Policer Parameters

To configure policing parameters, create a policer that specifies the maximum bandwidth and burst rate for traffic on an interface, and how to handle traffic that exceeds these values:

```
vSmart(config)# policy policer policer-name
vSmart(config-policer)# rate bps
vSmart(config-policer)# burst bytes
vSmart(config-policer)# exceed action
```

rate is the maximum traffic rate. It can be a value from 0 through $2^{64} - 1$ bits per second.

burst is the maximum traffic burst size. It can be a value from 15000 to 1000000 bytes

exceed is the action to take when the burst size or traffic rate is exceeded. *action* can be **drop** (the default) or **remark** . The **drop** action is equivalent to setting the packet loss priority (PLP) bit to low. The **remark** action sets the PLP bit to high. In centralized data policy, access lists, and application-aware routing policy, you can match the PLP with the **match plp** option.

Sequences

Each VPN list consists of sequences of match–action pairs. The sequences are numbered to set the order in which data traffic is analyzed by the match–action pairs in the policy. You configure sequences with the **policy data-policy vpn-list sequence** command.

Each sequence can contain one **match** command and one **action** command.

Match Parameters

Centralized data policy can match IP prefixes and fields in the IP headers, as well as applications. You can also enable split DNS. You configure the match parameters under the **policy data-policy vpn-list sequence match** command.

Each sequence in a policy can contain one **match** command.

For data policy, you can match these parameters:

Description	Command	Value or Range
Match all packets	Omit **match** command	—

Applications or application families	**app-list** *list-name*	Name of an **app-list** list
Group of destination prefixes	**destination-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual destination prefix	**destination-ip** *prefix / length*	IP prefix and prefix length
Destination port number	**destination-port** *number*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
Enable split DNS, to resolve and process DNS requests and responses on an application-by-application basis	**dns-app-list** *list-name*	Name of an **app-list** list. This list specifies the applications whose DNS requests are processed.
Specify the direction in which to process DNS packets	**dns** (**request** | **response**)	To process DNS requests sent by the applications (for outbound DNS queries), specify **dns request** . To process DNS responses returned from DNS servers to the applications, specify **dns response** .
DSCP value	**dscp** *number*	0 through 63
Packet length	**packet-length** *number*	0 through 65535; specify a single length, a list of lengths (with numbers separated by a space), or a range of lengths (with the two numbers separated with a hyphen [-])
Packet loss priority (PLP)	**plp**	(**high** | **low**) By default, packets have a PLP value of **low** . To set the PLP value to **high** , apply a policer that includes the **exceed remark** option.
Internet protocol number	**protocol** *number*	0 through 255
Group of source prefixes	**source-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual source prefix	**source-ip** *prefix / length*	IP prefix and prefix length
Source port number	**source-port** *address*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
TCP flag	**tcp** *flag*	**syn**

Action Parameters

When data traffic matches the conditions in the match portion of a centralized data policy, the packet can be accepted or dropped, and it can be counted. Then, you can associate parameters with accepted packets. You configure the action parameters under the **policy data-policy vpn-list sequence action** command.

Each sequence in a centralized data policy can contain on **action** command.

In the action, you first specify whether to accept or drop a matching data packet, and whether to count it:

Description	Command	Value or Range

Accept the packet. An accepted packet is eligible to be modified by the additional parameters configured in the **action** portion of the policy configuration.	**accept**	—
Enable cflowd traffic monitoring.	**cflowd**	—
Count the accepted or dropped packets.	**count** *counter-name*	Name of a counter. Use the **show policy access-lists counters** command on the vEdge router.
Discard the packet. This is the default action.	**drop**	—
Log the packet. Packets are placed into the messages and vsyslog system logging (syslog) files.	**log**	To view the packet logs, use the **show app log flows** and **show log** commands.
Redirect DNS requests to a particular DNS server. Redirecting requests is optional, but if you do so, you must specify both actions.	**redirect-dns host** **redirect-dns** *ip-address*	For an inbound policy, **redirect-dns host** allows the DNS response to be correctly forwarded back to the requesting service VPN. For an outbound policy, specify the IP address of the DNS server.
Fine-tune TCP to decrease round-trip latency and improve throughout for matching TCP traffic.	**tcp-optimization**	—

Then, for a packet that is accepted, the following parameters can be configured:

Description	Parameter	Value or Range
Enable cflowd traffic monitoring.	**cflowd**	—
Direct matching traffic to the NAT functionality so that it can be redirected directly to the Internet or other external destination.	**nat** [**pool** *number*] [**use-vpn 0**]	—
DSCP value.	**set dscp** *value*	0 through 63
Forwarding class.	**set forwarding-class** *value*	Name of forwarding class
Direct matching packets to a TLOC that mathces the color and encapsulation By default, if the TLOC is not available, traffic is forwarded using an alternate TLOC.	**set local-tloc color** *color* [**encap** *encapsulation*]	*color* can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . By default, *encapsulation* is **ipsec** . It can also be **gre** .
Direct matching packets to one of the TLOCs in the list if the TLOC matches the color and encapsulation By default, if the TLOC is not available, traffic is forwarded using an alternate TLOC. To drop traffic if a TLOC is unavailable, include the **restrict** option.	**set local-tloc-list color** *color* **encap** *encapsulation* [**restrict**]	*color* can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . By default, *encapsulation* is **ipsec** . It can also be **gre** .
Set the next hop to which the packet should be forwarded.	**set next-hop** *ip-address*	IP address
Apply a policer.	**set policer** *policer-name*	Name of policer configured with a **policy policer** command

Specify a service to redirect traffic to before delivering the traffic to its destination. The TLOC address or list of TLOCs identifies the remote TLOCs to which the traffic should be redirected to reach the service. In the case of multiple TLOCs, the traffic is load-balanced among them. The VPN identifier is where the service is located. Configure the services themselves on the vEdge routers that are collocated with the service devices, using the **vpn service** command.	**set service** *service-name* [**tloc** *ip-address* | **tloc-list** *list-name*] [**vpn** *vpn-id*]	Standard services: **FW** , **IDS** , **IDP** Custom services: **netsvc1** , **netsvc2** , **netsvc3** , **netsvc4** TLOC list is configured with a **policy lists tloc-list** list
Direct traffic to a remote TLOC that matches the IP address, color, and encapsulation.	**set tloc** *address* **color** *color* [**encap** *encapsulation*]	TLOC address, color, and encapsulation
Direct traffic to one of the remote TLOCs in the TLOC list if it matches the IP address, color, and encapsulation of one of the TLOCs in the list. If a preference value is configured for the matching TLOC, that value is assigned to the traffic.	**set tloc-list** *list-name*	Name of a **policy lists tloc-list** list
Set the VPN that the packet is part of.	**set vpn** *vpn-id*	0 through 65530

Default Action

If a data packet being evaluated does not match any of the match conditions in a data policy, a default action is applied to the packet. By default, the data packet is dropped. To modify this behavior, include the **policy data-policy vpn-list default-action accept** command.

Applying Centralized Data Policy

For a centralized data policy to take effect, you apply it to a list of sites in the overlay network:

vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* (**all** | **from-service** | **from-tunnel**)

By default, data policy applies to all data traffic passing through the vEdge router: the policy evaluates all data traffic going from the local site (that is, from the service side of the router) into the tunnel interface, and it evaluates all traffic entering to the local site through the tunnel interface. You can explicitly configure this behavior by including the **all** option. To have the data policy apply only to traffic coming from the service site and exiting from the local site through the tunnel interface, include the **from-service** option. To have the policy apply only to traffic entering from the tunnel interface and traveling to the service site, include the **from-tunnel** option. You can apply different data policies in each of the two traffic directions.

For all **data-policy** policies that you apply with **apply-policy** commands, the site IDs across all the site lists must be unique. That is, the site lists must not contain overlapping site IDs. An example of overlapping site IDs are those in the two site lists **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** . Here, sites 70 through 100 are in both lists. If you were to apply these two site lists to two different **data-policy** policies, the attempt to commit the configuration on the vSmart controller would fail.

The same type of restriction also applies to the following types of policies:

- Application-aware routing policy (**app-route-policy**)
- Centralized control policy (**control-policy**)
- Centralized data policy used for cflowd flow monitoring (**data-policy** hat includes a **cflowd** action and **apply-policy** that includes a **cflowd-template** command)

You can, however, have overlapping site IDs for site lists that you apply for different types of policy. For example, the sites lists for **control-policy** and **data-policy** policies can have overlapping site IDs. So for the two example site lists above, **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** , you could apply one to a control policy and the other to a data policy.

As soon as you successfully activate the configuration by issuing a **commit** command, the vSmart controller pushes the data policy to the vEdge routers located in the specified sites. To view the policy as configured on the vSmart controller, use the **show running-config** command on the vSmart controller:

```
vSmart# show running-config policy
vSmart# show running-config apply-policy
```

To view the policy that has been pushed to the vEdge router, use the **show policy from-vsmart** command on the vEdge router.

```
vEdge  # show policy from-vsmart
```

Configuring Deep Packet Inspection

You configure deep packet inspection using a standard centralized data policy. You define the applications of interest in a **policy lists app-list** command, and you call these lists in the **match** portion of the data policy. You can control the path of the application traffic through the network by defining, in the **action** portion of the data policy, the local TLOC or the remote TLOC, or for strict control, you can define both.

General vManage Configuration Procedure

To configure a centralized data policy for deep packet inspection in vManage NMS, perform the following steps:

1. Configure lists to group related items to be called in the centralized data policy.
2. Configure the centralized data policy.
3. Apply the policy.

Configure Lists

1. In vManage NMS, select the Configuration ► Policies screen.
2. In the Policies title bar, click the Centralized Policy/Localized Policy drop-down. When you first open the Policy Screen, Centralized Policy is selected by default.
3. Click Define Lists, located in the upper right corner of the screen.
4. In the left pane, select the type of list. For centralized data policy for deep packet inspection, you can use Application, Site, and VPN lists.
5. To create a new list, click New List.
   To modify an existing list, click the More Actions icon to the right of the desired list, and click the pencil icon.
6. In the List Name field, enter a name for the list. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
7. In the field below the List Name field, enter the desired values for the list. For some lists you type the desired values, and for others you select from a drop-down.
8. Click Add (for a new list) or Save (for an existing list).

Configure a Centralized Data Policy

1. In vManage NMS, select the Configuration ► Policies screen.
2. In the Policy title bar, click the Centralized Policy/Localized Policy drop-down. When you first open the Policy Screen, Centralized Policy is selected by default.
3. In the Policy bar, click Traffic.
4. To create a new centralized data policy, click Data Policy.
   To modify an existing policy, click the More Actions icon to the right of the desired policy, and click the pencil icon.
5. If data traffic does not match any of the conditions in one of the sequences, it is dropped by default. If you want nonmatching routes to be accepted, click the pencil icon in the Default Action, click Accept, and click Save Match And Actions.
6. To create a match–action sequence for data traffic:
   a. Click Sequence Type.
   b. To create a match–action rule, click Sequence Rule. The Match button is selected by default.
   c. Click the desired Match button, and enter the desired values in Match Conditions. For some conditions, you type the desired values, and for others you select from a drop-down.
   d. Click the Actions button. The default action is Reject. To accept matching packets, click the Accept radio button. Then click the desired action, and enter the desired values for Actions.
   e. Click Save Match and Actions.
   f. Create additional Sequence Rules or Sequence Types, as needed.
7. To rename a Sequence Type, double-click its name in the right pane, and type the new name. The name also changes in the right pane.
8. To re-order sequence rules and types, drag and drop them them.
9. Click Save.

You can also configure a centralized data policy for deep packet inspection directly from the Configuration ► Policies screen:

1. Click Assemble Full Policy.
2. In the Policy Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
3. In the Policy Description field, enter a description for the route policy. This field is mandatory, and it can contain any characters and spaces.
4. Click Data in the bar located directly below the Policy Description field.
5. In the left pane, click Add Data Policy, and follow Steps 6, 7, and 8 above.

Apply a Centralized Data Policy

1. In vManage NMS, select the Configuration ► Policies screen.
2. Click Assemble Full Policy.
3. In the Policy Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
4. In the Policy Description field, enter a description for the route policy. This field is mandatory, and it can contain any characters and spaces.

5. Click Data in the bar located directly below the Policy Description field.
6. In the left pane, select a data policy. The right pane displays the New Site List and VPN List box.
7. Click New Site List and VPN List.
8. Click the Select Site List field, and select a site list.
9. Click the Select VPN List field, and select a VPN list.
10. Click Add.
11. To add additional components to the centralized data policy, repeat Steps 6 through 10.
12. Click Save.

General CLI Configuration Procedure

Following are the high-level steps for configuring a centralized data policy to use for deep packet inspection:

1. Create a list of overlay network sites to which the data policy is to be applied (in the **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists- *list-name*)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).
   Create additional site lists, as needed.

2. Create lists of applications and application families that are to be subject to the data policy, Each list can contain one or more application names, or one or more application families. A single list cannot contain both applications and application families.
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **app-list** *list-name*
   vSmart(config-app-list)# **app** *application-name*

   vSmart(config)# **policy lists**
   vSmart(config-lists)# **app-list** *list-name*
   vSmart(config-applist)# **app-family** *family-name*

3. Create lists of IP prefixes and VPNs, as needed:
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **data-prefix-list** *list-name*
   vSmart(config-lists- *list-name*)# **ip-prefix** *prefix* / *length*

   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists- *list-name*)# **vpn** *vpn-id*

4. Create lists of TLOCs, as needed:
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists tloc-list** *list-name*
   vSmart(config-lists- *list-name*)# **tloc** *ip-address* **color** *color* **encap** *encapsulation* [**preference** *number*]

5. Define policing parameters, as needed:
   vSmart(config-policy)# **policer** *policer-name*
   vSmart(config-policer)# **rate** *bandwidth*
   vSmart(config-policer)# **burst** *bytes*
   vSmart(config-policer)# **exceed** *action*

6. Create a data policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy data-policy** *policy-name*
   vSmart(config-data-policy- *policy-name*)# **vpn-list** *list-name*

7. Create a series of match–pair sequences:
   vSmart(config-vpn-list)# **sequence** *number*
   vSmart(config-sequence- *number*)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

8. Define match parameters based on applications:
   vSmart(config-sequence- *number*)# **match app-list** *list-name*

9. Define additional match parameters for data packets:
   vSmart(config-sequence- *number*)# **match** *parameters*

10. Define actions to take when a match occurs:
    vSmart(config-sequence- *number*)# **action** (**accept** | **drop**) [**count**]

11. For packets that are accepted, define the actions to take. To control the tunnel over which the packets travels, define the remote or local TLOC, or for strict control over the tunnel path, set both:
    vSmart(config-action)# **set tloc** *ip-address* **color** *color* **encap** *encapsulation*
    vSmart(config-action)# **set tloc-list** *list-name*
    vSmart(config-action)# **set local-tloc color** *color* **encap** *encapsulation*
    vSmart(config-action)# **set local-tloc-list color** *color* **encap** *encapsulation* [**restrict**]

12. Define additional actions to take.

13. Create additional numbered sequences of match–action pairs within the data policy, as needed.

14. If a route does not match any of the conditions in one of the sequences, it is rejected by default. If you want nonmatching prefixes to be accepted, configure the default action for the policy:
    vSmart(config- *policy-name*)# **default-action accept**

15. Apply the policy to one or more sites in the overlay network:
    vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* (**all** | **from-service** | **from-tunnel**)

To enable the infrastructure for deep packet inspection on the vEdge routers, include the following command in the configuration on the routers:

vEdge(config)# **policy app-visibility**

Structural Components of Policy Configuration for Deep Packet Inspection

Following are the structural components required to configure centralized data policy for deep packet inspection. Each one is explained in more detail in the sections below.

```
On the vSmart controller:
policy
  lists
    app-list list-name
      (app applications | app-family application-families)
    data-prefix-list list-name
      ip-prefix prefix
    site-list list-name
      site-id site-id
    tloc-list list-name
      tloc ip-address color color encap encapsulation [preference value]
    vpn-list list-name
```

```
      vpn vpn-id
  policer policer-name
    burst bytes
    exceed action
    rate bps
  data-policy policy-name
    vpn-list list-name
      sequence number
        match
          app-list list-name
          destination-data-prefix-list list-name
          destination-ip ip-addresses
          destination-port port-numbers
          dscp number
          packet-length number
          protocol protocol
          source-data-prefix-list list-name
          source-ip ip-addresses
          source-port port-numbers
          tcp flag
        action
          drop
          count counter-name
          log
          accept
            nat [pool number] [use-vpn 0]
            set
              dscp number
              forwarding-class class
              local-tloc color color [encap encapsulation] [restrict]
              next-hop ip-address
              policer policer-name
              service service-name local [restrict] [vpn vpn-id]
              service service-name (tloc ip-address | tloc-list list-name) [vpn vpn-id]
              tloc ip-address color color encap encapsulation
              tloc-list list-name
              vpn vpn-id
      default-action
        (accept | drop)
apply-policy site-list list-name
  data-policy policy-name (all | from-service | from-tunnel)

On the vEdge router:
policy
  app-visibility
```

Lists

Centralized data policy for deep packet inspection uses the following types of lists to group related items. You configure lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command

Application list	List of one or more applications or application families running on the subnets connected to the vEdge router. • *application-names* can be the names of one or more applications. The Viptela software supports about 2300 different applications. To list the supported applications, use the **?** in the CLI. • *application-families* can be one or more of the following: **antivirus** , **application-service** , **audio_video** , **authentication** , **behavioral** , **compression** , **database** , **encrypted** , **erp** , **file-server** , **file-transfer** , **forum** , **game** , **instant-messaging** , **mail** , **microsoft-office** , **middleware** , **network-management** , **network-service** , **peer-to-peer** , **printer** , **routing** , **security-service** , **standard** , **telephony** , **terminal** , **thin-client** , **tunneling** , **wap** , **web** , and **webmail** .	**app-list** *list-name* (**app** *applications* | **app-family** *application-families*)
Prefix list	List of one or more IP prefixes.	**prefix-list** *list-name* **ip-prefix** *prefix* / *length*
Site list	List of one or more site identifiers in the overlay network. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*
TLOC list	List of one or more TLOCs in the overlay network. For each TLOC, specify its address, color, and encapsulation. *address* is the system IP address. *color* can be one of **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . *encapsulation* can be **gre** or **ipsec** . Optionally, set a preference value (from 0 to $2^{32} - 1$) to associate with the TLOC address. When you apply a TLOC list in an **action accept** condition, when multiple TLOCs are available and satisfy the match conditions, the TLOC with the lowest preference value is used. If two or more of TLOCs have the lowest preference value, traffic is sent among them in an ECMP fashion.	**tloc-list** *list-name* **tloc** *ip-address* **color** *color* **encap** *encapsulation* [**preference** *value*]
VPN list	List of one or more VPNs in the overlay network. For data policy, you can configure any VPNs except for VPN 0 and VPN 512. You can specify a single VPN identifier (such as **vpn 1**) or a range of VPN identifiers (such as **vpn 1-10**).	**vpn-list** *list-name* **vpn** *vpn-id*

In the vSmart controller configuration, you can create multiple iterations of each type of list. For example, it is common to create multiple site lists and multiple VPN lists so that you can apply data policy to different sites and different customer VPNs across the network.

You cannot apply the same type of policy to site lists that contain overlapping site IDs. That is, all data policies cannot have overlapping site lists among themselves. If you accidentally misconfigure overlapping site lists, the attempt to commit the configuration on the vSmart controller fails.

VPN Lists

Each centralized data policy is associated with a VPN list. You configure VPN lists with the **policy data-policy vpn-list** command. The list you specify must be one that you created with a **policy lists vpn-list** command.

Yo can include any VPNs except for VPN 0, which is reserved for control traffic, so never carries any data traffic, and VPN 512, which is reserved for out-of-band network management, so also never carries any data traffic. Note that while the CLI allows you to include these two VPNs in a data policy configuration, the policy is not applied to these two VPNs.

Policer Parameters

To configure policing parameters, create a policer that specifies the maximum bandwidth and burst rate for traffic on an interface, and how to handle traffic that exceeds these values:

```
vSmart(config)# policy policer policer-name
vSmart(config-policer)# rate bps
```

```
vSmart(config-policer)# burst bytes
vSmart(config-policer)# exceed action
```

rate is the maximum traffic rate. It can be a value from 0 through $2^{64} - 1$ bits per second.

burst is the maximum traffic burst size. It can be a value from 15000 to 1000000 bytes

exceed is the action to take when the burst size or traffic rate is exceeded. *action* rop (the default) or **remark** . The **drop** action is equivalent to setting the packet loss priority (PLP) bit to low. The **remark** action sets the PLP bit to high. In centralized data policy, access lists, and application-aware routing policy, you can match the PLP with the **match plp** option.

Sequences

Within each VPN list are sequences of match–action pairs. The sequences are numbered to set the order in which data traffic is analyzed by the match–action pairs in the policy. You configure sequences with the **policy data-policy vpn-list sequence** command.

Each sequence can contain one **match** command and one **action** command.

Match Parameters

For deep packet inspection, centralized data policy must match one or more applications. It can also match IP prefixes and fields in the IP headers. You configure the match parameters under the **policy data-policy vpn-list sequence match** command.

Each sequence in a policy can contain one **match** command.

For data policy, you can match these parameters:

Description	Command	Value or Range
Group of applications	**app-list** *list-name*	Name of an **app-list** list
Group of destination prefixes	**destination-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual destination prefix	**destination-ip** *prefix / length*	IP prefix and prefix length
Destination port number	**destination-port** *number*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
DSCP value	**dscp** *number*	0 through 63
Packet length	**packet-length** *number*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
Internet Protocol number	**protocol** *number*	0 through 255
Group of source prefixes	**source-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual source prefix	**source-ip** *prefix / length*	IP prefix and prefix length

Source port number	**source-port** *address*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
TCP flag	**tcp** *flag*	**syn**

Action Parameters

When data traffic matches the conditions in the match portion, the data packets can be accepted or dropped, and they can be counted. Then, you can associate parameters with accepted packets. You configure the action parameters under the **policy data-policy vpn-list sequence action** command.

Each sequence in a centralized data policy can contain one **action** command.

In the action, you first specify whether to accept or drop a matching data packet, and whether to count it:

Description	Command	Value or Range
Accept the packet. An accepted packet is eligible to be modified by the additional parameters configured in the **action** portion of the policy configuration.	**accept**	—
Count the accepted or dropped packets.	**count** *counter-name*	Name of a counter. Use the **show policy access-lists counter** command on the vEdge router to display counter information.
Discard the packet. This is the default action.	**drop**	—
Place a sampled set of packets that match the match conditions into the messages and vsyslog system logging (syslog) files.	**log**	—

To view the packet logs, use the **show app log flows** and **show log** commands.

Then, for a packet that is accepted, the following parameters can be configured. Note that you cannot use DPI with either cflowd or NAT.

Description	Parameter	Value or Range
DSCP value.	**set dscp** *value*	0 through 63
Forwarding class.	**set forwarding-class** *value*	Name of forwarding class
Direct matching packets to a TLOC that mathces the color and encapsulation By default, if the TLOC is not available, traffic is forwarded using an alternate TLOC.	**set local-tloc color** *color* [**encap** *encapsulation*]	*color* can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . By default, *encapsulation* is **ipsec** . It can also be **gre** .
Direct matching packets to one of the TLOCs in the list if the TLOC matches the color and encapsulation By default, if the TLOC is not available, traffic is forwarded using an alternate TLOC. To drop traffic if a TLOC is unavailable, include the **restrict** option.	**set local-tloc-list color** *color* **encap** *encapsulation* [**restrict**]	*color* can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , and **silver** . By default, *encapsulation* is **ipsec** . It can also be **gre** .

Set the next hop to which matching packets should be forwarded.	**set next-hop** *ip-address*	IP address.
Apply a policer.	**set policer** *policer-name*	Name of policer configured with a **policy policer** command
Direct matching packets to the name service, before delivering the traffic to its ultimate destination. The TLOC address or list of TLOCs identifies the remote TLOCs to which the traffic should be redirected to reach the service. In the case of multiple TLOCs, the traffic is load-balanced among them. The VPN identifier is where the service is located. Configure the services themselves on the vEdge routers that are collocated with the service devices, using the **vpn service** configuration command.	**set service** *service-name* [**tloc** *ip-address* | **tloc-list** *list-name*] [**vpn** *vpn-id*]	Standard services: **FW** , **IDS** , **IDP** Custom services: **netsvc1** , **netsvc2** , **netsvc3** , **netsvc4** TLOC list is configured with a **policy lists tloc-list** command
Direct matching packets to the named service that is reachable via a GRE tunnel whose source is in the transport VPN (VPN 0). If the GRE tunnel used to reach the service is down, packet routing falls back to using standard routing. To drop packets when a GRE tunnel to the service is unreachable, include the **restrict** option. In the service VPN, you must also advertise the service using the **service** command. You configure the GRE interface or interfaces in the transport VPN (VPN 0).	**set service** *service-name* **local** [**restrict**] [**vpn** *vpn-id*]	Standard services: **FW** , **IDS** , **IDP** Custom services: **netsvc1** , **netsvc2** , **netsvc3** , **netsvc4**
Direct traffic to a remote TLOC. The TLOC is defined by its IP address, color, and encapsulation.	**set tloc** *address* **color** *color* [**encap** *encapsulation*]	TLOC address, color, and encapsulation
Direct traffic to one of the remote TLOCs in the TLOC list.	**set tloc-list** *list-name*	Name of a **policy lists tloc-lists** list
Set the VPN that the packet is part of.	**set vpn** *vpn-id*	0 through 65530

Default Action

If a data packet being evaluated does not match any of the match conditions in a control policy, a default action is applied to this route. By default, the data packet is dropped. To modify this behavior, include the **policy data-policy vpn-list default-action accept** command.

Applying Centralized Data Policy for Deep Packet Inspection

For a deep packet inspection centralized data policy to take effect, you apply it to a list of sites in the overlay network:

vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* (**all** | **from-service** | **from-tunnel**)

By default, data policy applies to all data traffic passing through the vEdge router: the policy evaluates all data traffic going from the local site (that is, from the service side of the router) into the tunnel interface, and it evaluates all traffic entering to the local site through the tunnel interface. You can explicitly configure this behavior by including the **all** option. To have the data policy apply only to policy exiting from the local site, include the **from-service** option. To have the policy apply only to incoming traffic, include the **from-tunnel** option.

You cannot apply the same type of policy to site lists that contain overlapping site IDs. That is, all data policies cannot have overlapping site lists among themselves. If you accidentally misconfigure overlapping site lists, the attempt to commit the configuration on the vSmart controller fails.

As soon as you successfully activate the configuration by issuing a **commit** command, the vSmart controller pushes the data policy to the vEdge routers located in the specified sites. To view the policy as configured on the vSmart controller, use the **show running-config** command on the vSmart controller:

vSmart# **show running-config policy**
vSmart# ; **show running-config apply-policy**

To view the policy that has been pushed to the vEdge router, use the **show policy from-vsmart** command on the vEdge router.

vEdge# **show policy from-vsmart**

Monitor Running Applications

To enable the deep packet inspection infrastructure on the vEdge routers, you must enable application visibility on the routers:

```
vEdge(config)# policy app-visibility
```

To display information about the running applications, use the **show app dpi supported-applications** , **show app dpi applications** , and **show app dpi flows** commands on the router.

Configuring VPN Membership Policy

A VPN membership data policy consists of a series of numbered (ordered) sequences of match-action pair that are evaluated in order, from lowest sequence number to highest sequence number. When a packet matches one of the match conditions, the associated action is taken and policy evaluation on that packets stops. Keep this in mind as you design your policies to ensure that the desired actions are taken on the items subject to policy.

If a packet matches no parameters in any of the sequences in the policy configure, it is, by default, rejected and discarded.

To create a VPN membership policy, you include the following components in the configuration on a vSmart controller:

Component	Description	Configuration Command
Lists	Groupings of related items that you reference in the match and action portions of the data policy configuration. For VPN membership policy, you can group sites and VPNs.	**policy lists**
Centralized VPN membership policy instance	Container for VPN membership policy to filter packets based on VPN.	**policy vpn-membership**
Numbered sequences of match–action pairs	Sequences that establish the order in which the policy components are applied	**policy vpn-membership sequence**
Match parameters	Conditions that packets must match to be considered for the VPN membership policy.	**policy vpn-membership sequence match**
Actions	Whether to accept or reject matching packets.	**policy vpn-membership sequence action**
Default action	Action to take if a packet matches none of the policy conditions.	**policy vpn-membership default-action**
Application of VPN membership policy	For a VPN membership policy to take effect, you apply it to one or more sites in the overlay network.	**apply-policy site-list vpn-membership**

General Configuration Procedure

Following are the high-level steps for configuring a VPN membership data policy:

1. Create a list of overlay network sites to which the VPN membership policy is to be applied (in the **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists- *list-name*)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).
   Create additional site lists, as needed.

2. Create lists VPNs, as needed:
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists- *list-name*)# **vpn** *vpn-id*

3. Create a VPN membership policy instance:
   vSmart(config)# **policy vpn-membership** *policy-name*

4. Create a series of match–pair sequences:
   vSmart(config- *policy-nsme*)# **sequence** *number*
   vSmart(config-sequence- *number*)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the packet or accepting it).

5. Define match parameters for VPNs:
   vSmart(config-sequence-number)# **match** (**vpn** *vpn-id* | **vpn-list** *list-name*)

6. Define actions to take when a match occurs:
   vSmart(config-sequence- *number*)# **action** (**accept** | **reject**)

7. Create additional numbered sequences of match–action pairs within the data policy, as needed.

8. If a packet does not match any of the conditions in one of the sequences, it is rejected by default. If you want nonmatching packets to be accepted, configure the default action for the policy:
   vSmart(config- *policy-name*)# **default-action accept**

9. Apply the policy to one or more sites in the overlay network:
   vSmart(config)# **apply-policy site-list** *list-name* **vpn-membership** *policy-name*

Structural Components of VPN Membership Policy Configuration

Following are the structural components required to configure VPN membership policy. Each one is explained in more detail in the sections that follow.

```
policy
  lists
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn vpn-id
  vpn-membership policy-name
    sequence number
      match
        match-parameters
      action
        (accept | reject)
      default-action
```

```
          (accept | reject)
apply-policy site-list list-name
  vpn-membership policy-name
```

Lists

Centralized data policy uses the following types of lists to group related items. You configure lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command
Site list	List of site identifiers in the overlay network. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*
VPN list	List of VPNs in the overlay network	**vpn-list** *list-name* **vpn** *vpn-id*

In the vSmart controller configuration, you can create multiple iterations of each type of list. For VPN membership policy, you commonly create multiple site lists and multiple VPN lists so that you can apply data policy to different sites and different customer VPNs across the network.

You cannot apply the same type of policy to site lists that contain overlapping site IDs. That is, all VPN membership policies cannot have overlapping site lists among themselves. If you accidentally misconfigure overlapping site lists, the attempt to commit the configuration on the vSmart controller fails.

Match Parameters

For VPN membership policy, you can match these parameters:

Description	Command	Value or Range
Individual VPN identifier	**vpn** *vpn-id*	0 through 65535
Name of a VPN list.	**vpn-list** *list-name*	Name of a **vpn-list** name

Action Parameters

When data traffic matches the conditions in the match portion of a VPN membership policy, the packet can be either accepted or rejected:

Description	Command	Value or Range
Accept the packet.	**accept**	—
Reject the packet. This is the default action.	**reject**	—

Default Action

If a data packet being evaluated does not match any of the match conditions in a VPN membership policy, a default action is applied to this route. By default, the route is rejected. To modify this behavior, include the **default-action accept** command in the VPN membership policy.

Applying VPN Membership Policy

For a VPN membership policy to take effect, you must apply it to a list of sites in the overlay network:

```
vSmart(config)# apply-policy site-list list-name vpn-membership policy-name
```

You cannot apply the same type of policy to site lists that contain overlapping site IDs. That is, all data policies cannot have overlapping site lists among themselves. If you accidentally misconfigure overlapping site lists, the attempt to commit the configuration on the vSmart controller fails.

To display the VPN membership policy as configured on the vSmart controller, use the **show running-config** command on the vSmart controller:

```
vSmart# show running-config policy
vSmart# show running-config apply-policy
```

Additional Information

Application-Aware Routing
Centralized Data Policy

Centralized Data Policy Configuration Examples

This article provides some straightforward examples of configuring centralized data policy to influence traffic flow across the Viptela domain and to configure a vEdge router to be an Internet exit point.

General Centralized Data Policy Example

This section shows a general example of a centralized data policy to illustrate that you configure centralized data policy on a vSmart controller and that after you commit the configuration, the policy itself is pushed to the affected vEdge routers.

Here we configure a simple data policy on the vSmart controller vm9:

```
vm9# show running-config policy
policy
 data-policy test-data-policy
  vpn-list test-vpn-list
   sequence 10
    match
     destination-ip 172.16.0.0/24
    !
    action drop
     count test-counter
    !
   !
   default-action drop
  !
 !
 lists
  vpn-list test-vpn-list
   vpn 1
  !
  site-list test-site-list
   site-id 500
  !
 !
!
```

Then we apply this policy to the site list named **test-site-list** , which includes site 500:

```
vm9# show running-config apply-policy
apply-policy
 site-list test-site-list
  data-policy test-data-policy
 !
!
```

Immediately after we activate the configuration on the vSmart controller, it pushes the policy configuration to the vEdge routers in site 500. One of these routers is vm5, where we see that the policy has been received:

```
vm5# show omp data-policy
policy-from-vsmart
 data-policy test-data-policy
  vpn-list test-vpn-list
   sequence 10
    match
     destination-ip 172.16.0.0/24
    !
    action drop
     count test-counter
    !
   !
   default-action drop
  !
 !
 lists
  vpn-list test-vpn-list
   vpn 1
  !
 !
!
```

Control Access

This example shows a data policy that limits the type of packets that a source can send to a specific destination. Here, the host at source address 1.1.1.1 in site 100 and VPN 100 can send only TCP traffic to the destination host at 2.2.2.2. This policy also specifies the next hop for the TCP traffic sent by 1.1.1.1, setting it to be TLOC 10.10.10.10, color gold. All other traffic is accepted as a result of the **default-action** statement.

```
policy
  lists
     site-list north
       site-id 100
     vpn-list vpn-north
       vpn 100
  !
  data-policy tcp-only
     vpn-list vpn-north
       sequence 10
         match
           source-ip 1.1.1.1/32
           destination-ip 2.2.2.2/32
           protocol tcp
         action accept
           set tloc 10.10.10.10 color gold
       !
       default-action accept
   !
!
apply-policy
   site north data-policy tcp-only
```

Restrict Traffic

This examples illustrates how to disallow certain types of data traffic from being sent from between VPNs. This policy drops data traffic on port 25, which carries SMTP mail traffic, that originates in 1.1.0.0/16. However, the policy accepts all other data traffic, including non-SMTP traffic from 1.1.0.0/16.

```
policy
  lists
    data-prefix-list north-ones
      ip-prefix 1.1.0.0/16
      port 25
    vpn-list all-vpns
      vpn 1
      vpn 2
    site-list north
      site-id 100
  !
  data-policy no-mail
   vpn-list all-vpns
     sequence 10
       match
         source-data-prefix-list north-ones
       action drop
     !
     default-action accept
  !
!
apply-policy
  site north data-policy no-mail
```

Allow Traffic to Exit from a vEdge Router to the Internet

The following example allows data traffic destined for two prefixes on the Internet to exit directly from the local vEdge router to the Internet destination. Configure this policy on the vSmart controller.

```
policy
 lists
  vpn-list vpn-1
    vpn 1
  !
  site-list nat-sites
    site-id 100,200
  !
data-policy accept-nat
  vpn-list vpn-1
   sequence 100
    match
     source-ip      10.20.24.0/24
     destination-ip 10.0.12.12/32
    !
    action accept
     count nat
     nat use-vpn 0
    !
   !
   sequence 101
    match
     source-ip      10.20.24.0/24
     destination-ip 10.1.15.13/32
    !
    action accept
     count nat_inet
```

```
      nat use-vpn 0
     !
    !
    default-action accept
   !
 !
apply-policy
  site-list nat-sites data-policy accept-nat
```

Using the destination port instead of a destination IP prefix allows greater flexibility for traffic exiting to the Internet. Here, traffic can go to all HTTP and HTTPS sites (ports 80 and 443, respectively). Configure this policy on a vSmart controller.

```
data-policy accept-nat
  vpn-list vpn-1
   sequence 100
    match
     source-ip      10.20.24.0/24
     destination-port 80
    !
    action accept
     count nat
     nat use-vpn 0
    !
   !
   sequence 101
    match
     source-ip      10.20.24.0/24
     destination-port 443
    !
    action accept
     count nat_inet
     nat use-vpn 0
    !
   !
   default-action accept
  !
 !
```

Additional Information

Centralized Data Policy
Configuring Centralized Data Policy

Localized Data Policy

Data policy operates on the data plane in the Viptela overlay network and affects how data traffic is sent among the vEdge routers in the network. The Viptela architecture defines two types of data policy, centralized data policy, which controls the flow of data traffic based on the IP header fields in the data packets and based on network segmentation, and localized data policy, which controls the flow of data traffic into and out of interfaces and interface queues on a vEdge router.

Localized data policy, so called because it is provisioned on the local vEdge router, is applied on a specific router interface and affects how a specific interface handles the data traffic that it is transmitting and receiving. Localized data policy is also referred to as access lists (ACLs). With access lists, you can provision class of service (CoS), classifying data packets and prioritizing the transmission properties for different classes. You can also provision packet mirroring and policing. For IPv4, you you also configure QoS actions.

You can apply IPv4 access lists in any VPN on the router, and you can create access lists that act on unicast and multicast traffic. You can apply IPv6 access lists only to tunnel interfaces in the transport VPN (VPN 0).

You can apply access lists either in the outbound or inbound direction on the interface. Applying an IPv4 ACL in the outbound direction affects data packets traveling from the local service-side network into the IPsec tunnel toward the remote service-side network. Applying an IPv4 ACL in the inbound direction affects data packets exiting from the IPsec tunnel and being received by the local vEdge router. For IPv6, an outbound ACL is applied to traffic being transmitted by the router, and an inbound ACL is applied to received traffic.

Explicit and Implicit Access Lists

Access lists that you configure using localized data policy are called *explicit* ACLs. You can apply explicit ACLs in any VPN on the router.

Router tunnel interfaces also have *implicit ACLs* , which are also referred to as *services* . Some of these are present by default on the tunnel interface, and they are in effect unless you disable them. Through configuration, you can also enable other implicit ACLs. On vEdge routers, the following services are enabled by default: DHCP (for DHCPv4 and DHCPv6), DNS, and ICMP. You can also enable services for BGP, Netconf, NTP, OSPF, SSHD, and STUN.

Perform QoS Actions

With access lists, you can provision quality of service (QoS) which allows you to classify data traffic by importance, spread it across different interface queues, and control the rate at which different classes of traffic are transmitted. See **Forwarding and QoS Overview.**

Mirror Data Packets

Once packets are classified, you can configure access lists to send a copy of data packets seen on a vEdge router interface to a specified destination on another network device. The Viptela software supports 1:1 mirroring; that is, a copy of every packet is sent to the alternate destination.

Additional Information

Centralized Data Policy
Configuring Localized Data Policy for IPv4
Configuring Localized Data Policy for IPv6
Forwarding and QoS Overview
Forwarding and QoS Configuration Examples

Configuring Localized Data Policy for IPv4

This article provides procedures for configuring IPv4 localized data policy This type of data policy is called access lists, or ACLs. You can provision simple access lists that filter traffic based on IP header fields. You also use access lists to apply QoS, mirroring, and policing to data packets. You can create access lists that act on unicast and multicast traffic.

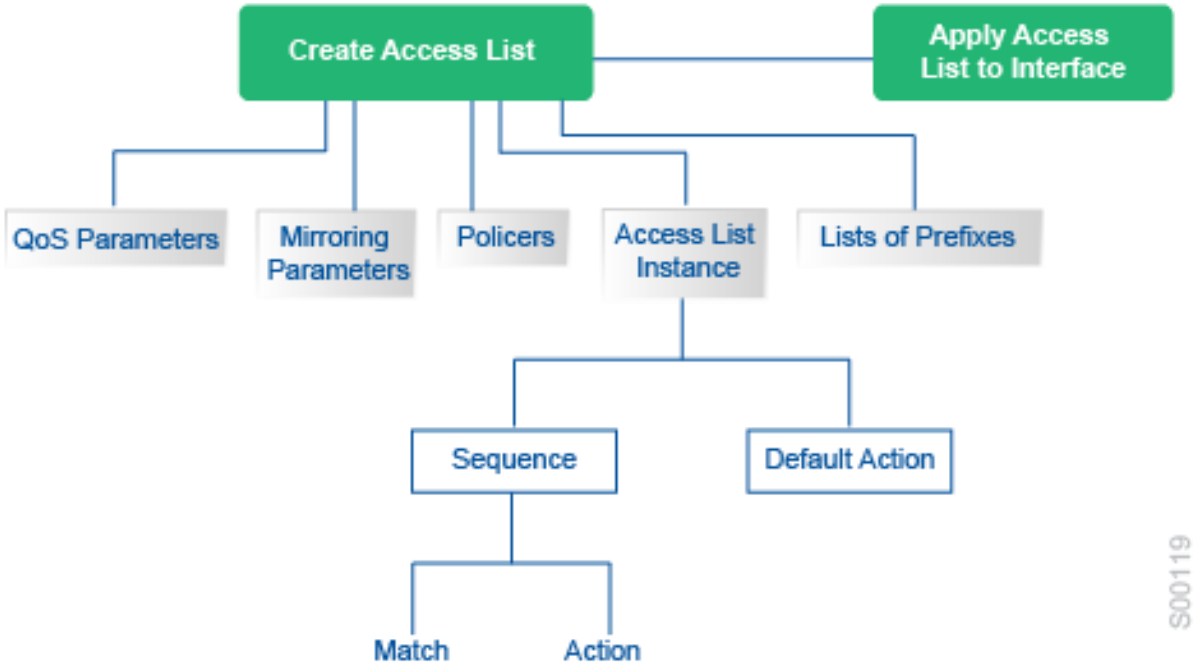
In vManage NMS, you configure localized data policy from the Configuration ► Policies screen, using a policy configuration wizard. In the CLI you configure these policies on the vEdge router.

Configuration Components

An access list consists of a sequences of match–action pairs that are evaluated in order, from lowest sequence number to highest sequence number. When a packet matches one of the match conditions, the associated action is taken and policy evaluation on that packets stops. Keep this in mind as you design your policies to ensure that the desired actions are taken on the items subject to policy.

If a packet matches no parameters in any of the sequences in the policy configuration, it is, by default, dropped.

The following figure illustrates the configuration components for access lists.



To create an access list, you include the following components in the configuration:

Component	Description	vManage Configuration	CLI Configuration Command
Lists	Groupings of related items that you reference in the match and action portions of the data policy configuration.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Create Groups of Interest or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Create Groups of Interest	**policy lists**
Logging frequency	If you configure a logging action, log only a sample of data packet headers instead of all of them.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Policy Overview ▶ Log Frequency	**policy logging-frequency**

QoS, mirroring, and policing parameters	Parameters and rules required to configure QoS, traffic mirroring, and traffic policing. For QoS, you can configure class maps, QoS maps, the QoS scheduler, and rewrite rules. For mirroring, you configure the addresses of the source of the packets to be mirrored and the mirroring site. (You can mirror only unicast traffic.) For policing, you define transmission parameters.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Forwarding Classes/QoS, Policy Overview or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Forwarding Class/QoS Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Create Groups of Interest or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Create Groups of Interest	**policy class-map** **policy cloud-qos** **policy-cloud-qos-service-side** **policy qos-scheduler** **policy qos-map** **policy rewrite-rule** **policy mirror** **policy policer**
Access list instance	Container for an access list.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy access-list**
Numbered sequences of match–action pairs	Sequences establish the order in which the policy components are applied.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy access-list sequence**
Match parameters	Conditions that packets must match to be considered for a data policy.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy access-list sequence match**

Actions	Whether to accept or reject matching packets, and how to process matching items.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy access-list sequence action**
Default action	Action to take if a packet matches none of the match parameters in any of the sequences. By default, nonmatching packets are dropped.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy access-list default-action**
Application of access lists	For an access list to take effect, you apply it an interface. You can also apply policers directly to interfaces.	Configuration ▶ Templates ▶ Feature ▶ VPN Interface Bridge, VPN Interface Cellular, VPN Interface Ethernet, VPN Interface GRE, VPN Interface PPP, or VPN Interface PPP Ethernet	**vpn interface access-list** **vpn interface policer**

General vManage Configuration Procedure

To configure IPv4 localized policy, use the vManage policy configuration wizard. The wizard is a UI policy builder that consists of five screens to configure IPv4 localized policy components:

- Groups of Interest, also called lists—Create data prefix lists and mirroring and policer parameters that group together related items and that you call in the match or action components of a policy.
- Forwarding Classes—Define forwarding classes and rewrite rules to use for QoS.
- Access Control Lists—Define the match and action conditions of ACLs.
- Route Policies—Define the match and action conditions of route policies.
- Policy Settings—Define additional policy settings, including Cloud QoS settings and the frequency for logging policy-related packet headers.

You configure some or all these components depending on the specific policy you are creating. To skip a component, click the Next button at the bottom of the screen. To return to a component, click the Back button at the bottom of the screen.

Start the Policy Configuration Wizard

To start the policy configuration wizard:

1. In vManage NMS, select the Configure ▶ Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Select the Localized Policy tab.
3. Click Add Policy.

The policy configuration wizard opens, and the Create Groups of Interest screen is displayed.

Create Groups of Interest

To create lists of groups to use in localized data policy:

1. Start the policy configuration wizard as explained above.
2. Create new lists, as described in the following table:

List Type	Procedure
Data Prefix	1. In the left bar, click Data Prefix. 2. Click New Data Prefix List. 3. Enter a name for the list. 4. Enter one or more IP prefixes. 5. Click Add.
Mirror	6. In the left bar, click TLOC. 7. Click New TLOC List. The TLOC List popup displays. 8. Enter a name for the list. 9. In the TLOC IP field, enter the system IP address for the TLOC. 10. In the Color field, select the TLOC's color. 11. In the Encap field, select the encapsulation type. 12. In the Preference field, optionally select a preference to associate with the TLOC. 13. Click Add TLOC to add another TLOC to the list. 14. Click Save.
Policer	15. In the left bar, click VPN. 16. Click New VPN List. 17. Enter a name for the list. 18. In the Add VPN field, enter one or more VPN IDs separated by commas. 19. Click Add.

4. Click Next to move to Configure Forwarding Classes/QoS in the wizard.

Configure Forwarding Classes for QoS

When you first open the Forwarding Classes/QoS screen, the QoS tab is selected by default.

To configure forwarding classes for use by QoS:

1. To create a new QoS mapping:

a. In the QoS tab, click the Add QoS drop-down.

b. Select Create New.

c. Enter a name and description for the QoS mapping.

d. Click Add Queue. The Add Queue popup displays.

e. Select the queue number from the Queue drop-down.

f. Select the maximum bandwidth and buffer percentages, and the scheduling and drop types. Enter the forwarding class.

g. Click Save.

2. To import an existing QoS mapping:

a. In the QoS tab, click the Add QoS drop-down.

b. Select Import Existing.

c. Select a QoS mapping.

d. Click Import.

3. To view or copy a QoS mapping or to remove the mapping from the localized policy, click the More Actions icon to the right of the row, and select the desired action.

4. To configure policy rewrite rules for the QoS mapping:

a. In the QoS tab, click the Add Rewrite Policy drop-down..

b. Select Create New.

c. Enter a name and description for the rewrite rule.

d. Click Add Rewrite Rule. The Add Rule popup displays.

e. Enter the class name and DSCP value, and select the priority level.

f. Click Save.

5. To import an existing rewrite rule:

a. In the QoS tab, click the Add Rewrite Policy drop-down..

b. Select Import Existing.

c. Select a rewrite rule.

d. Click Import.

6. Click Next to move to Configure Access Lists in the wizard.

Configure ACLs

To configure access control lists (ACLs):

1. To create a new IPv4 ACL, click the Add Access Control List Policy drop-down. Then select Add IPv4 ACL Policy.

2. Enter a name and description for the ACL.

3. In the left pane, click Add ACL Sequence. An Access Control List box is displayed in the left pane.

4. Double-click the Access Control List box, and type a name for the ACL.
5. In the right pane, click Add Sequence Rule to create a single sequence in the ACL. The Match tab is selected by default.
6. Click a match condition.
7. On the left, enter the values for the match condition.
8. On the right enter the action or actions to take if the policy matches.
9. Repeat Steps 6 through 8 to add match–action pairs to the ACL.
10. To rearrange match–action pairs in the ACL, in the right pane drag them to the desired position.
11. To remove a match–action pair from the ACL, click the X in the upper right of the condition.
12. Click Save Match and Actions to save a sequence rule.
13. To rearrange sequence rules in an ACL, in the left pane drag the rules to the desired position.
14. To copy, delete, or rename an ACL sequence rule, in the left pane, click More Options next to the rule's name and select the desired option.
15. If no packets match any of the ACL sequence rules, the default action is to drop the packets. To change the default action:
    a. Click Default Action in the left pane.
    b. Click the Pencil icon.
    c. Change the default action to Accept.
    d. Click Save Match and Actions.
16. Click Next to move to Configure Route Policy in the wizard. Click Next to move to the Policy Overview screen.

Configure Policy Settings

To configure policy settings, in the Policy Overview screen:

1. Enter a name and description for the route policy.
2. To enable QoS scheduling and shaping for traffic that a vEdge Cloud router receives from transport-side interfaces, click Cloud QoS.
3. To enable QoS scheduling and shaping for traffic that a vEdge Cloud router receives from service-side interfaces, click Cloud QoS Service Side.
4. To log the headers of all packets that are dropped because they do not match a service configured by an Allow Service parameter on a tunnel interface, click Implicit ACL Logging.
5. To configure how often packets flows are logged, click Log Frequency. Packet flows are those that match an access list (ACL), a cflowd flow, or an application-aware routing flow.
6. Click Preview to view the full policy in CLI format.
7. Click Save Policy.

Apply a Localized Data Policy in a Device Template

1. n vManage NMS, select the Configuration ► Templates screen.
2. If you are creating a new device template:
    a. In the Device tab, click Create Template.
    b. From the Create Template drop-down, select From Feature Template.
    c. From the Device Model drop-down, select one of the vEdge devices.
    d. In the Template Name field, enter a name for the device template. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
    e. In the Description field, enter a description for the device template. This field is mandatory, and it can contain any characters and spaces.
    f. Continue with Step 4.
3. If you are editing an existing device template:
    a. In the Device tab, click the More Actions icon to the right of the desired template, and click the pencil icon.
    b. Click the Additional Templates tab. The screen scrolls to the Additional Templates section.
    c. From the Policy drop-down, select the name of a policy that you have configured.
4. Click the Additional Templates tab located directly beneath the Description field. The screen scrolls to the Additional Templates section.
5. From the Policy drop-down, select the name of the policy you configured in the above procedure.
6. Click Create (for a new template) or Update (for an existing template).

General CLI Configuration Procedure

Following are the high-level steps for configuring an access list using the CLI:

1. Create lists of IP prefixes, as needed:
   vEdge(config)# **policy**
   vEdge(config-policy)# **lists** **data-prefix-list** *list-name*
   vEdge(config-data-prefix-list)# **ip-prefix** *prefix / length*
2. If you configure a logging action, configure how often to log packets to the syslog files:
   vEdge(config)# **policy** **log-frequency** *number*
3. For QoS, map each forwarding class to an output queue, configure a QoS scheduler for each forwarding class, and group the QoS schedulers into a QoS map:
   vEdge(config)# **policy** **class-map**
   vEdge(config-class-map)# **class** *class-name* **queue** *number*

   vEdge(config)# **policy** **qos-scheduler** *scheduler-name*
   vEdge(config-qos-scheduler)# **class** *class-name*
   vEdge(config-qos-scheduler)# **bandwidth-percent** *percentage*
   vEdge(config-qos-scheduler)# **buffer-percent** *percentage*
   vEdge(config-qos-scheduler)# **drops** *drop-type*
   vEdge(config-qos-scheduler)# **scheduling** *type*

   vEdge(config)# **policy qos-map** *map-name* **qos-scheduler** *scheduler-name*

4. For QoS, define rewrite rules to overwrite the DSCP field of a packet's outer IP header, if desired:
   vEdge(config)# **policy rewrite-rule** *rule-name*
   vEdge(config-rewrite-rule)# **class** *class-name loss-priority* **dscp** *dscp-value*
   class-name is one of the classes defined under a **qos-scheduler** command.

5. Define mirroring parameters (for unicast traffic only):
   vEdge(config)# **policy mirror** *mirror-name*
   vEdge(config-mirror)# **remote-dest** *ip-address* **source** *ip-address*

6. Define policing parameters:
   vEdge(config)# **policy policer** *policer-name*
   vEdgeconfig-policer)# **rate** *bandwidth*
   vEdge(config-policer)# **burst** *bytes*
   vEdge(config-policer)# **exceed** *action*

7. Create an access list instance:
   vEdge(config)# **policy access-list** *list-name*

8. Create a series of match–action pair sequences:
   vEdge(config-access-list)# **sequence** *number*
   vEdge(config-sequence)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

9. Define match parameters for packets:
   vEdge(config-sequence- *number*)# **match** *match- parameter*

10. Define actions to take when a match occurs:
    vEdge (config-sequence)# **action drop**
    vEdge(config-sequence)# **action count** *counter-name*
    vEdge(config-sequence)# **action log**
    vEdge(config-sequence)# **action accept class** *class-name*
    vEdge(config-sequence)# **action accept mirror** *mirror-name*
    vEdge(config-sequence)# **action accept policer** *policer-name*
    vEdge(config-sequence)# **action accept set dscp** *value*
    vEdge(config-sequence)# **action accept set next-hop** *ipv4-address*

11. Create additional numbered sequences of match–action pairs within the access list, as needed.

12. If a packet does not match any of the conditions in one of the sequences, it is rejected by default. If you want nonmatching packets to be accepted, configure the default action for the access list:
    vEdge(config- *policy-name*)# **default-action accept**

13. Apply the access list to an interface:
    vEdge(config)# **vpn** *vpn-id* **interface** *interface-name*
    vEdge(config-interface)# **access-list** *list-name* (**in** | **out**)
    Applying the access list in the inbound direction (**in**) affects packets being received on the interface. Applying it in the outbound direction (**out**) affects packets being transmitted on the interface.
    For QoS, apply a DSCP rewrite rule to the same egress interface:
    vEdge(config)# **vpn** *vpn-id* **interface** *interface-name* **rewrite-rule** *rule-name*
    Note that it is also possible to apply a policerdirectly to an interface, which has the effect of policing all packets transiting the interface, rather than policing only the selected packets that match the access list. You can apply the policer to either inbound or outbound packets:
    vEdge(config)# **vpn** *vpn-id* **interface** *interface-name*
    vEdge(config-interface)# **policer** *policer-name* (**in** | **out**)

Structural Components of Configuration for Access Lists

Following are the structural components required to configure access lists, shown as they appear in the CLI and when you click Preview in the vManage localized policy configuration wizard. Each component is explained in the sections below.

```
policy
  lists
    data-prefix-list list-name
      ip-prefix prefix/length
  class-map
    class class map map
  cloud-qos
  cloud-qos-service-side
  implicit-acl-logging
  log-frequency number
  qos-scheduler scheduler-name
    class class-name
    bandwidth-percent percentage
    buffer-percent percentage
    drops drop-type
    scheduling (llq | wrr)
  qos-map map-name
    qos-scheduler scheduler-name
  rewrite-rule rule-name
    class class-name priority dscp dscp-value
  mirror mirror-name
    remote-dest ip-address source ip-address
  policer policer-name
    rate bandwidth
    burst bytes
    exceed action
  access-list list-name
    sequence number
      match
        match-parameters
      action
        drop
          count counter-name
          log
        accept
          class class-name
          count counter-name
          log
          mirror mirror-name
          policer policer-name
          set dscp value
          set next-hop ipv4-address
    default-action
      (accept | drop)
vpn vpn-id
  interface interface-name
    access-list list-name (in | out)
    policer policer-name (in | out)
    rewrite-rule rule-name
```

Lists

Access lists use prefix lists to group related prefixes. You configure lists under the **policy lists** command hierarchy on vEdge routers.

List Type	Description	Command

Data prefixes	List of one or more IP prefixes. You can specify both unicast and multicast addresses. To configure multiple prefixes in a single list, include multiple **ip-prefix** options, specifying one prefix in each option.	**data-prefix-list** *list-name* **ip-prefix** *prefix / length*

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Create Groups of Interest ► Data Prefix ► New Data Prefix List
- Configuration ► Policies ► Custom Options ► Localized Policy ► Lists ► Data Prefix ► New Data Prefix List

Logging Parameters

If you configure a logging action in a data policy, by default, the vEdge router logs all data packet headers to a syslog file. To log only a sample of the data packet headers:

```
vEdge(config)# policy log-frequency number
```

number specifies how often to to log packet headers. The default value is 1000. *number* can be an integer., and the software rounds the value down to the nearest power of 2. So for example, with the default value of 1000, the logging frequency is rounded down to 512, so every 512th packet is logged.

You can log the headers of all packets that are dropped because they do not match a service configured with an **allow-service** command. You can use these logs for security purposes, for example, to monitor the flows that are being directed to a WAN interface and to determine, in the case of a DDoS attack, which IP addresses to block.

```
vEdge(config)# policy implicit-acl-logging
```

When you enable implicit ACL logging, by default, the headers of all dropped packets are logged. It is recommended that you configure a limit to the number of packets logged with the **log-frequency** command.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Policy Overview ► Log Frequency field
- Configuration ► Policies ► Localized Policy ► Add Policy ► Policy Overview ► Implicit ACL Logging field

QoS Parameters

To configure QoS parameters on a vEdge router, first define a classification:

```
vEdge(config)# policy class-map class class-name queue number
```

class-name is the name of the class. It can be a text string from 1 through 32 characters long.

For hardware vEdge routers, each interface has eight queues, numbered from 0 through 7. Queue 0 is reserved for low-latency queuing (LLQ), so any class that is mapped to queue 0 must be configured to use LLQ. The default scheduling method for all is weighted round-robin (WRR).

For Cloud vEdge routers, each interface has four queues, numbered from 0 through 3. Queue 0 is reserved for control traffic, and queues 1, 2, and 3 are available for data traffic. The scheduling method for all four queues is WRR. LLQ is not supported.

To configure QoS parameters on a vEdge Cloud router, you must enable QoS scheduling and shaping. To enable QoS parameters for traffic that the vEdge Cloud router receives from transport-side interfaces:

```
vEdgeCloud(config)# policy cloud-qos
```

To enable QoS parameters for traffic that the vEdge Cloud router receives from service-side interfaces:

```
vEdgeCloud(config)# policy cloud-qos-service-side
```

Next, configure scheduling:

```
vEdge(config)# policy qos-scheduler scheduler-name
vEdge(config-qos-scheduler)# class class-name
vEdge(config-qos-scheduler)# bandwidth-percent percentage
vEdge(config-qos-scheduler)# buffer-percent percentage
vEdge(config-qos-scheduler)# drops (red-drop | tail-drop)
vEdge(config-qos-scheduler)# scheduling (llq | wrr)
```

scheduler-name is the name of the QoS scheduler. It can be a text string from 1 through 32 characters long.

class-name is the name of the forwarding class and can be a text string from 1 through 32 characters long. The common class names correspond to the per-hop behaviors AF (assured forwarding), BE (best effort), and EF (expedited forwarding).

The bandwidth percentage is the percentage of the interface's bandwidth to allocate to the forwarding class. The sum of the bandwidth on all forwarding classes on an interface should not exceed 100 percent.

The buffer percentage is the percentage of the interface's buffering capacity to allocate to the forwarding class. The sum of the buffering capacity of all forwarding classes on an interface should not exceed 100 percent.

Packets that exceed the bandwidth or buffer percentage are dropped either randomly, using random early detection (**red-drop**), or from the end of the queue (**tail-drop**). Low-latency queuing (LLQ) cannot use random early detection.

The algorithm to schedule interface queues can be either low-latency queuing (**llq**) or weighted round-robin (**wrr**).

Then, assign the scheduler to a QoS map:

```
vEdge(config-policy)# qos-map map-name qos-scheduler scheduler-name
```

map-name is the name of the QoS map, and *scheduler-name* is the name of the scheduler you configured above. Each name& can be a text string from 1 through 32 characters long.

Finally, to configure a rewrite rule to overwrite the DSCP field of a packet's outer IP header:

```
vEdge(config)# policy rewrite-rule rule-name class class-name loss-priority dscp dscp-value
```

rule-name is the name of the rewrite rule. It can be a text string from 1 through 32 characters long.

class-name is the name of a class you configured with the **qos-scheduler class** command. The packet loss priority (PLP) can be either **high** or **low** . The DSCP value to overwrite the DSCP field of the packet's outer IP header can be from 0 through 63.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Forwarding Classes/QoS ► QoS Map ► Add QoS Map
- Configuration ► Policies ► Custom Options ► Localized Policy ► Forwarding Classes/QoS ► QoS Map ► Add QoS Map
- Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Forwarding Classes/QoS ► Policy Rewrite ► Add Rewrite Policy
- Configuration ► Policies► Custom Options ► Localized Policy ► Forwarding Classes/QoS ► Policy Rewrite ► Add Rewrite Policy

Mirroring Parameters

To configure mirroring parameters, define the remote destination to which to mirror the packets, and define the source of the packets:

```
vEdge(config)# policy mirror mirror-name
vEdge(config-mirror)# remote-dest ip-address source ip-address
```

Mirroring applies to unicast traffic only. It does not apply to multicast traffic.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Create Groups of Interest ► Mirror ► New Mirror List

- Configuration ► Policies ► Custom Options ► Localized Policy ► Lists ► Mirror ► New Mirror List

Policer Parameters

To configure policing parameters, create a policer that specifies the maximum bandwidth and burst rate for traffic on an interface, and how to handle traffic that exceeds these values:

```
vEdge(config)# policy policer policer-name
vEdge(config-policer)# rate bps
vEdge(config-policer)# burst bytes
vEdge(config-policer)# exceed action
```

rate is the maximum traffic rate. It can be a value from 0 through $2^{64} - 1$ bits per second.

burst is the maximum traffic burst size. It can be a value from 15000 to 1000000 bytes

exceed is the action to take when the burst size or traffic rate is exceeded. *action* can be **drop** (the default) or **remark** . The **drop** action is equivalent to setting the packet loss priority (PLP) bit to low. The **remark** action sets the PLP bit to high. In centralized data policy, access lists, and application-aware routing policy, you can match the PLP with the **match plp** option.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Create Groups of Interest ► Policer ► New Policer List
- Configuration ► Policies ► Custom Options ► Localized Policy ► Lists ► Policer ► New Policer List

Sequences

An access list contains sequences of match–action pairs. The sequences are numbered to set the order in which a packet is analyzed by the match–action pairs in the access lists. You configure sequences with the **policy access-list sequence** command.

Each sequence in an access list can contain one **match** command and one **action** command.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Access Control Lists ► Add Access Control List Policy ► Add ACL Sequence
- Configuration ► Policies ► Custom Options ► Localized Policy ► Access Control List Policy ► Add Access Control List Policy ► Add ACL Sequence

Match Parameters

Access lists can match IP prefixes and fields in the IP headers. You configure the match parameters under the **policy access-list sequence match** command.

Each sequence in an access-list must contain one **match** command.

For access lists, you can match these parameters:

Description	Command	Value or Range
Classification map	**class** *class-name*	Name of a class defined with a **policy class-map** command.
Group of destination prefixes	**destination-data-prefix-list** *list-name*	Name of a **data-prefix-list** list.

Individual destination prefix	**destination-ip** *prefix / length*	IP prefix and prefix length
Destination port number .	**destination-port** *number*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
DSCP value	**dscp** *number*	0 through 63
Internet Protocol number	**protocol** *number*	0 through 255
Packet length	**packet-length** *number*	Length of the packet. *number* can be from 0 through 65535. Specify a single length, a list of lengths (with numbers separated by a space), or a range of lengths (with the two numbers separated with a hyphen [-])
Group of source prefixes	**source-data-prefix-list** *list-name*	Name of a **data-prefix-list** list.
Packet loss priority (PLP)	**plp**	(**high** | **low**) By default, packets have a PLP value of **low** . To set the PLP value to **high** , apply a policer that includes the **exceed remark** option.
Individual source prefix	**source-ip** *prefix / length*	IP prefix and prefix length
Source port number .	**source-port** address	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
TCP flag	**tcp** *flag*	**syn**

In vManage NMS:

- Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists ▶ Add Access Control List Policy ▶ Add ACL Sequence ▶ Add Sequence Rule ▶ Match
- Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control List Policy ▶ Add Access Control List Policy ▶ Add ACL Sequence ▶ Add Sequence Rule ▶ Match

Action Parameters

When a packet matches the conditions in the match portion of an access list, the packet can be accepted or dropped, and it can be counted. Then, you can classify, mirror, or police accepted packets. You configure the actions parameters with the **policy access-list sequence action** command.

Each sequence in an access list can contain one **action** command.

In the action, you first specify whether to accept or drop a matching data packet, and whether to count it:

Description	Command	Value or Range
Accept the packet. An accepted packet is eligible to be modified by the additional parameters configured in the **action** portion of the access list.	**accept**	—
Count the accepted or dropped packets.	**count** *counter-name*	Name of a counter. To display counter information, use the **show policy access-lists counters** command on the vEdge router.
Discard the packet. This is the default action.	**drop**	—

Log the packet headers into the messages and vsyslog system logging (syslog) files. In addition to logging the packet headers, a syslog message is generated the first time a packet header is logged and then every 5 minutes thereafter, as long as the flow is active.	**log**	To display logging information, use the **show app log flow-all** , **show app log flows** , and **show log** commands on the vEdge router.

For a packet that is accepted, the following actions can be configured:

Description	Command	Value or Range
Classify the packet.	**class** *class-name*	Name of a QoS class defined with a **policy class-map** command.
Mirror the packet.	**mirror** *mirror-name*	Name of mirror defined with a **policy mirror** command.
Police the packet.	**police** *policer-name*	Name of a policer defined with a **policy policer** command.
Packet's DSCP value.	**set dscp** *value*	0 through 63.
Next-hop address.	**set next-hop** *ipv4-address*	IPv4 address.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Access Control Lists ► Add Access Control List Policy ► Add ACL Sequence ► Add Sequence Rule& ► Action
- Configuration ► Policies ► Custom Options ► Localized Policy ► Access Control List Policy ► Add Access Control List Policy ► Add ACL Sequence ► Add Sequence Rule ► Action

Default Action

If a packet being evaluated does not match any of the match conditions in a access list, a default action is applied to this packet. By default, the packet is dropped. To modify this behavior, include the **access-list default-action accept** command in the access list.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Access Control Lists ► Default Action
- Configuration ► Policies ► Custom Options ► Localized Policy ► Access Control List Policy ► Default Action

Apply Access Lists

For an access list to take effect, you must apply it to an interface:

```
vEdge(config)# vpn vpn-id interface interface-name
vEdge(config-interface)# access-list list-name (in | out)
```

Applying the policy in the inbound direction (**in**) affects prefixes being received on the interface. Applying it in the outbound direction (**out**) affects prefixes being transmitted on the interface.

For an access list that applies QoS classification, apply any DSCP rewrite rules to the same interface to which you apply the access list:

```
vEdge(config)# vpn vpn-id interface interface-name rewrite-rule rule-name
```

Note that you can also apply a policer directly to an interface, which has the effect of policing all packets transiting the interface, rather than policing only the selected packets that match the access list. You can apply the policer to either inbound or outbound packets:

```
vEdge(config)# vpn vpn-id interface interface-name
vEdge(config-interface)# policer policer-name (in | out) interface-name
```

In vManage NMS:

- Configuration ▶ Templates ▶ VPN Interface Bridge
- Configuration ▶ Templates ▶ VPN Interface Cellular
- Configuration ▶ Templates ▶ VPN Interface Ethernet
- Configuration ▶ Templates ▶ VPN Interface GRE
- Configuration ▶ Templates ▶ VPN Interface PPP
- Configuration ▶ Templates ▶ VPN Interface PPP Ethernet

Interaction between Explicit and Implicit Access Lists

Access lists that you configure through localized data policy using the **policy access-list** command are called *explicit* ACLs. You can apply explicit ACLs to any interface in any VPN on the router.

The router's tunnel interfaces in VPN 0 also have *implicit ACLs* , which are also referred to as *services* . Some services are enabled by default on the tunnel interface, and are in effect unless you disable them. Through configuration, you can also enable other services. You configure and modify implicit ACLs with the **allow-service** command:

```
vEdge(config)# vpn 0
vEdge(config-vpn)# interface interface-name
vEdge(config-interface)# tunnel-interface
vEdge(config-tunnel-interface)# allow-service service-name
vEdge(config-tunnel-interface)# no allow-service service-name
```

On vEdge routers, the following services are enabled by default: DHCP (for DHCPv4 and DHCPv6), DNS, and ICMP. These three services allow the tunnel interface to accept DHCP, DNS, and ICMP packets. You can also enable services for BGP, Netconf, NTP, OSPF, SSHD, and STUN.

When data traffic matches both an explicit ACL and an implicit ACL, how the packets are handled depends on the ACL configuration. Specifically, it depends on:

- Whether the implicit ACL is configured as allow (**allow-service** *allow-service*) or deny (**no allow-service** *service-name*). Allowing a service in an implicit ACL is the same as specifying the **accept** action in an explicit ACL, and a service that is not allowed in an implicit ACL is the same as specifying the **drop** action in an explicit ACL
- Whether, in an explicit ACL, the **accept** or **deny** action is configured in a policy sequence or in the default action.

The following table explains how traffic matching both an implicit and an explicit ACL is handled:

Implicit ACL	Explicit ACL: Sequence	Explicit ACL: Default	Result
Allow (accept)	Deny (drop)	—	Deny (drop)
Allow (accept)	—	Deny (drop)	Allow (accept)
Deny (drop)	Allow (accept)	—	Allow (accept)
Deny (drop)	—	Allow (accept)	Deny (drop)

Additional Information

Configuring Localized Data Policy for IPv6
Localized Data Policy

Configuring Localized Data Policy for IPv6

This article provides procedures for configuring IPv6 localized data policy This type of data policy is called access lists, or ACLs. You can provision simple access lists that filter traffic based on IP header fields. You also use access lists to apply mirroring and policing to data packets.

For IPv6, you can apply access lists only to interfaces in the transport VPN, VPN 0.

In vManage NMS, you configure localized data policy from the Configuration ► Policies screen, using a policy configuration wizard. In the CLI you configure these policies on the vEdge router.

Configuration Components

An access list consists of a sequences of match–action pairs that are evaluated in order, from lowest sequence number to highest sequence number. When a packet matches one of the match conditions, the associated action is taken and policy evaluation on that packets stops. Keep this in mind as you design your policies to ensure that the desired actions are taken on the items subject to policy.

If a packet matches no parameters in any of the sequences in the policy configuration, it is, by default, dropped.

To create an IPv6 access list, you include the following components in the configuration on a vEdge router:

Component	Description	vManage Configuration	CLI
Logging Frequency	If you configure a logging action, log only a sample of data packet headers instead of all of them.	Configuration ► Policies ► Localized Policy ► Add Policy ► Policy Overview ► Log Frequency	**policy logging-frequency**
Mirroring and policing parameters	Parameters and rules required to configure traffic mirroring and policing. For mirroring, you configure the addresses of the source of the packets to be mirrored and the mirroring site. (You can mirror only unicast traffic.) For policing, you define transmission parameters.	Configuration ► Policies ► Localized Policy ► Add Policy ► Create Groups of Interest or Configuration ► Policies ► Custom Options ► Localized Policy ► Create Groups of Interest	**policy mirror** **policy policer**
Access list instance	Container for an access list.	Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Access Control Lists or Configuration ► Policies ► Custom Options ► Localized Policy ► Access Control Lists	**policy ipv6 access-list**
Numbered sequences of match–action pairs	Sequences establish the order in which the policy components are applied.	Configuration ► Policies ► Localized Policy ► Add Policy ► Configure Access Control Lists or Configuration ► Policies ► Custom Options ► Localized Policy ► Access Control Lists	**policy ipv6 access-list sequence**

Match parameters	Conditions that packets must match to be considered for a data policy.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy ipv6 access-list sequence match**
Actions	Whether to accept or reject matching packets, and how to process matching items.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy ipv6 access-list sequence action**
Default action	Action to take if a packet matches none of the match parameters in any of the sequences. By default, nonmatching packets are dropped.	Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists or Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control Lists	**policy ipv6 access-list default-action**
Application of access lists	For an access list to take effect, you apply it an interface.	Configuration ▶ Templates ▶ Feature ▶ VPN Interface Bridge, VPN Interface Cellular, VPN Interface Ethernet, VPN Interface GRE, VPN Interface PPP, or VPN Interface PPP Ethernet	**vpn 0 interface ipv6 access-list**

General vManage Configuration Procedure

To configure IPv6 localized data policy, use the vManage policy configuration wizard. The wizard is a UI policy builder that consists of five screens, and you use four of them to configure IPv6 localized policy components:

- Groups of Interest, also called lists—Create data prefix lists and mirroring and policer parameters that group together related items and that you call in the match or action components of a policy.
- Access Control Lists—Define the match and action conditions of ACLs.
- Route Policies—Define the match and action conditions of route policies.
- Policy Settings—Define additional policy settings, including the frequency for logging policy-related packet headers.

You configure some or all these components depending on the specific policy you are creating. To skip a component, click the Next button at the bottom of the screen. To return to a component, click the Back button at the bottom of the screen.

Start the Policy Configuration Wizard

To start the policy configuration wizard:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Select the Localized Policy tab.
3. Click Add Policy. The policy configuration wizard opens, and the Create Groups of Interest screen is displayed.

Create Groups of Interest

To create lists of groups to use in localized data policy:

1. Start the policy configuration wizard as explained above.
2. Create new lists, as described in the following table:

List Type	Procedure
Mirror	1. In the left bar, click TLOC. 2. Click New TLOC List. The TLOC List popup displays. 3. Enter a name for the list. 4. In the TLOC IP field, enter the system IP address for the TLOC. 5. In the Color field, select the TLOC's color. 6. In the Encap field, select the encapsulation type. 7. In the Preference field, optionally select a preference to associate with the TLOC. 8. Click Add TLOC to add another TLOC to the list. 9. Click Save.
Policer	10. In the left bar, click VPN. 11. Click New VPN List. 12. Enter a name for the list. 13. In the Add VPN field, enter one or more VPN IDs separated by commas. 14. Click Add.

4. Click Next to move to Configure Forwarding Classes/QoS in the wizard. For IPv6 localized data policy, you cannot configure QoS.
5. Click Next to move to Configure Access Lists in the wizard.

Configure ACLs

To configure access control lists (ACLs):

1. To create a new IPv6 ACL, click the Add Access Control List Policy drop-down. Then select Add IPv6 ACL Policy.
2. Enter a name and description for the ACL.
3. In the left pane, click Add ACL Sequence. An Access Control List box is displayed in the left pane.

4. Double-click the Access Control List box, and type a name for the ACL.
5. In the right pane, click Add Sequence Rule to create a single sequence in the ACL. The Match tab is selected by default.
6. Click a match condition.
7. On the left, enter the values for the match condition.
8. On the right enter the action or actions to take if the policy matches.
9. Repeat Steps 6 through 8 to add match–action pairs to the ACL.
10. To rearrange match–action pairs in the ACL, in the right pane drag them to the desired position.
11. To remove a match–action pair from the ACL, click the X in the upper right of the condition.
12. Click Save Match and Actions to save a sequence rule.
13. To rearrange sequence rules in an ACL, in the left pane drag the rules to the desired position.
14. To copy, delete, or rename an ACL sequence rule, in the left pane, click More Options next to the rule's name and select the desired option.
15. If no packets match any of the ACL sequence rules, the default action is to drop the packets. To change the default action:
    a. Click Default Action in the left pane.
    b. Click the Pencil icon.
    c. Change the default action to Accept.
    d. Click Save Match and Actions.
16. Click Next to move to Configure Route Policy in the wizard. Click Next to move to the Policy Overview screen.

Configure Policy Settings

To configure policy settings, in the Policy Overview screen:

1. Enter a name and description for the route policy.
2. To log the headers of all packets that are dropped because they do not match a service configured by an Allow Service parameter on a tunnel interface, click Implicit ACL Logging.
3. To configure how often packets flows are logged, click Log Frequency. Packet flows are those that match an access list (ACL), a cflowd flow, or an application-aware routing flow.
4. Click Preview to view the full policy in CLI format.
5. Click Save Policy.

Apply a Localized Data Policy in a Device Template

1. n vManage NMS, select the Configuration ► Templates screen.
2. If you are creating a new device template:
    a. In the Device tab, click Create Template.

b. From the Create Template drop-down, select From Feature Template.

c. From the Device Model drop-down, select one of the vEdge devices.

d. In the Template Name field, enter a name for the device template. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.

e. In the Description field, enter a description for the device template. This field is mandatory, and it can contain any characters and spaces.

f. Continue with Step 4.

3. If you are editing an existing device template:

a. In the Device tab, click the More Actions icon to the right of the desired template, and click the pencil icon.

b. Click the Additional Templates tab. The screen scrolls to the Additional Templates section.

c. From the Policy drop-down, select the name of a policy that you have configured.

4. Click the Additional Templates tab located directly beneath the Description field. The screen scrolls to the Additional Templates section.

5. From the Policy drop-down, select the name of the policy you configured in the above procedure.

6. Click Create (for a new template) or Update (for an existing template).

General CLI Configuration Procedure

Following are the high-level steps for configuring an access list using the CLI:

1. Define mirroring parameters (for unicast traffic only):

2. vEdge(config)# **policy mirror** *mirror-name*
vEdge(config-mirror)# **remote-dest** *ip-address* **source** *ip-address*

3. Define policing parameters:
vEdge(config)# **policy policer** *policer-name*
vEdgeconfig-policer)# **rate** *bandwidth*
vEdge(config-policer)# **burst** *bytes*
vEdge(config-policer)# **exceed** *action*

4. Create an access list instance:
vEdge(config)# **policy ipv6 access-list** *list-name*

5. Create a series of match–action pair sequences:
vEdge(config-ipv6-access-list)# **sequence** *number*
vEdge(config-sequence)#
The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

6. Define match parameters for packets:
vEdge(config-sequence- *number*)# **match** *match- parameter*

7. Define actions to take when a match occurs:
vEdge(config-sequence)# **action drop**
vEdge(config-sequence)# **action count** *counter-name*
vEdge(config-sequence)# **action log**
vEdge(config-sequence)# **action accept class** *class-name*

vEdge(config-sequence)# **action accept mirror** *mirror-name*
vEdge(config-sequence)# **action accept policer** *policer-name*

8. Create additional numbered sequences of match–action pairs within the access list, as needed.

9. If a packet does not match any of the conditions in one of the sequences, it is rejected by default. If you want nonmatching packets to be accepted, configure the default action for the access list:
vEdge(config- *policy-name*)# **default-action accept**

10. Apply the access list to an interface:
vEdge(config)# **vpn** *vpn-id* **interface** *interface-name*
vEdge(config-interface)# **ipv6 access-list** *list-name* (**in** | **out**)
Applying the access list in the inbound direction (**in**) affects packets being received on the interface. Applying it in the outbound direction (**out**) affects packets being transmitted on the interface.

Structural Components of Configuration for Access Lists

Following are the structural components required to configure access lists. Each one is explained in more detail in the sections below.

```
policy
  implicit-acl-logging
  log-frequency number
  mirror mirror-name
    remote-dest ip-address source ip-address
  policer policer-name
    rate bandwidth
    burst bytes
    exceed action
policy ipv6
  access-list list-name
    sequence number
      match
        match-parameters
      action
        drop
        count counter-name
        log
        accept
          class class-name
          mirror mirror-name
          policer policer-name
    default-action
      (accept | drop)
vpn vpn-id
  interface interface-name
    ipv6 access-list list-name (in | out)
```

Logging Parameters

If you configure a logging action in a data policy, by default, the vEdge router logs all data packet headers to a syslog file. To log only a sample of the data packet headers:

```
vEdge(config)# policy log-frequency number
```

number specifies how often to to log packet headers. The default value is 1000. *number* can be an integer., and the software rounds the value down to the nearest power of 2. So for example, with the default value of 1000, the logging frequency is rounded down to 512, so every 512th packet is logged.

You can log the headers of all packets that are dropped because they do not match a service configured with an **allow-service** command. You can use these logs for security purposes, for example, to monitor the flows that are being directed to a WAN interface and to determine, in the case of a DDoS attack, which IP addresses to block.

```
vEdge(config)# policy implicit-acl-logging
```

When you enable implicit ACL logging, by default, the headers of all dropped packets are logged. It is recommended that you configure a limit to the number of packets logged with the **log-frequency** command.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Policy Overview ► Log Frequency field
- Configuration ► Policies ► Localized Policy ► Add Policy ► Policy Overview ► Implicit ACL Logging field

Mirroring Parameters

To configure mirroring parameters, define the remote destination to which to mirror the packets, and define the source of the packets:

```
vEdge(config)# policy mirror mirror-name
vEdge(config-mirror)# remote-dest ip-address source ip-address
```

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Create Groups of Interest ► Mirror ► New Mirror List
- Configuration ► Policies ► Custom Options ► Localized Policy ► Lists ► Mirror ► New Mirror List

Policer Parameters

To configure policing parameters, create a policer that specifies the maximum bandwidth and burst rate for traffic on an interface, and how to handle traffic that exceeds these values:

```
vEdge(config)# policy policer policer-name
vEdge(config-policer)# rate bps
vEdge(config-policer)# burst bytes
vEdge(config-policer)# exceed action
```

rate is the maximum traffic rate. It can be a value from 8 through 10000000 bits per second.

burst is the maximum traffic burst size. It can be a value from 15000 to 1000000 bytes

exceed is the action to take when the burst size or traffic rate is exceeded. *action* can be **drop** (the default) or **remark** . The **drop** action is equivalent to setting the packet loss priority (PLP) bit to low. The **remark** action sets the PLP bit to high. In centralized data policy, access lists, and application-aware routing policy, you can match the PLP with the **match plp** option.

In vManage NMS:

- Configuration ► Policies ► Localized Policy ► Add Policy ► Create Groups of Interest ► Policer ► New Policer List
- Configuration ► Policies ► Custom Options ► Localized Policy ► Lists ► Policer ► New Policer List

Sequences

An access list contains sequences of match–action pairs. The sequences are numbered to set the order in which a packet is analyzed by the match–action pairs in the access lists. You configure sequences with the **policy ipv6 access-list sequence** command.

Each sequence in an access list can contain one **match** command and one **action** command.

In vManage NMS:

- Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists ▶ Add Access Control List Policy ▶ Add ACL Sequence
- Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control List Policy ▶ Add Access Control List Policy ▶ Add ACL Sequence

Match Parameters

Access lists can match IP prefixes and fields in the IP headers. You configure the match parameters under the **policy access-list sequence match** command.

Each sequence in an access-list must contain one **match** command.

For access lists, you can match these parameters:

Description	Command	Value or Range
Destination port number .	**destination-port** *number*	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
Next header protocol	**next-header** *number*	0 through 255, corresponding to an Internet Protocol number
Packet length	**packet-length** *number*	Length of the packet. *number* can be from 0 through 65535. Specify a single length, a list of lengths (with numbers separated by a space), or a range of lengths (with the two numbers separated with a hyphen [-])
Packet loss priority (PLP)	**plp**	(**high** | **low**) By default, packets have a PLP value of **low** . To set the PLP value to **high** , apply a policer that includes the **exceed remark** option.
Source port number .	**source-port** address	0 through 65535; specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
TCP flag	**tcp** *flag*	**syn**
Traffic class	**traffic-class** *value*	0 through 63

In vManage NMS:

- Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists ▶ Add Access Control List Policy ▶ Add ACL Sequence ▶ Add Sequence Rule ▶ Match
- Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control List Policy ▶ Add Access Control List Policy ▶ Add ACL Sequence ▶ Add Sequence Rule ▶ Match

Action Parameters

When a packet matches the conditions in the match portion of an access list, the packet can be accepted or dropped, and it can be counted. Then, you can classify, mirror, or police accepted packets. You configure the actions parameters with the **policy access-list sequence action** command.

Each sequence in an access list can contain one **action** command.

In the action, you first specify whether to accept or drop a matching data packet, and whether to count it:

Description	Command	Value or Range

Accept the packet. An accepted packet is eligible to be modified by the additional parameters configured in the **action** portion of the access list.	**accept**	—
Count the accepted or dropped packets.	**count** *counter-name*	Name of a counter. To display counter information, use the **show ipv6 policy access-lists counters** command on the vEdge router.
Discard the packet. This is the default action.	**drop**	—
Log the packet headers into system logging (syslog) files. In addition to logging the packet headers, a syslog message is generated the first time a packet header is logged and then every 5 minutes thereafter, as long as the flow is active.	**log**	To display logging information, use the **show app log flow-all** and **show app log flows** command on the vEdge router.

For a packet that is accepted, the following actions can be configured:

Description	Command	Value or Range
Mirror the packet.	**mirror** *mirror-name*	Name of mirror defined with a **policy mirror** command.
Police the packet.	**police** *policer-name*	Name of a policer defined with a **policy policer** command.
Set the packet's DSCP value.	**set traffic-class** *value*	0 through 63.

In vManage NMS:

- Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists ▶ Add Access Control List Policy ▶ Add ACL Sequence ▶ Add Sequence Rule ▶ Action
- Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control List Policy ▶ Add Access Control List Policy ▶ Add ACL Sequence ▶ Add Sequence Rule ▶ Action

Default Action

If a packet being evaluated does not match any of the match conditions in a access list, a default action is applied to this packet. By default, the packet is dropped. To modify this behavior, include the **access-list ipv6 default-action accept** command in the access list.

In vManage NMS:

- Configuration ▶ Policies ▶ Localized Policy ▶ Add Policy ▶ Configure Access Control Lists ▶ Default Action
- Configuration ▶ Policies ▶ Custom Options ▶ Localized Policy ▶ Access Control List Policy ▶ Default Action

Applying Access Lists

For an access list to take effect, you must apply it to a tunnel interface in VPN 0:

```
vEdge(config)# vpn 0 interface interface-name
vEdge(config-interface)# ipv6 access-list list-name (in | out)
```

Applying the policy in the inbound direction (**in**) affects prefixes being received on the interface. Applying it in the outbound direction (**out**) affects prefixes being transmitted on the interface.

In vManage NMS:

- Configuration ▶ Templates ▶ VPN Interface Bridge

- Configuration ▶ Templates ▶ VPN Interface Cellular
- Configuration ▶ Templates ▶ VPN Interface Ethernet
- Configuration ▶ Templates ▶ VPN Interface GRE
- Configuration ▶ Templates ▶ VPN Interface PPP
- Configuration ▶ Templates ▶ VPN Interface PPP Ethernet

Interaction between Explicit and Implicit Access Lists

Access lists that you configure through localized data policy using the **policy access-list** command are called *explicit* ACLs. You can apply explicit ACLs to any interface in any VPN on the router.

The router's tunnel interfaces in VPN 0 also have *implicit ACLs* , which are also referred to as *services* . Some services are enabled by default on the tunnel interface, and are in effect unless you disable them. Through configuration, you can also enable other services. You configure and modify implicit ACLs with the **allow-service** command:

```
vEdge(config)# vpn 0
vEdge(config-vpn)# interface interface-name
vEdge(config-interface)# tunnel-interface
vEdge(config-tunnel-interface)# allow-service service-name
vEdge(config-tunnel-interface)# no allow-service service-name
```

On vEdge routers, the following services are enabled by default: DHCP (for DHCPv4 and DHCPv6), DNS, and ICMP. These three services allow the tunnel interface to accept DHCP, DNS, and ICMP packets. You can also enable services for BGP, Netconf, NTP, OSPF, SSHD, and STUN.

When data traffic matches both an explicit ACL and an implicit ACL, how the packets are handled depends on the ACL configuration. Specifically, it depends on:

- Whether the implicit ACL is configured as allow (**allow-service** *allow-service*) or deny (**no allow-service** *service-name*). Allowing a service in an implicit ACL is the same as specifying the **accept** action in an explicit ACL, and a service that is not allowed in an implicit ACL is the same as specifying the **drop** action in an explicit ACL
- Whether, in an explicit ACL, the **accept** or **deny** action is configured in a policy sequence or in the default action.

The following table explains how traffic matching both an implicit and an explicit ACL is handled:

Implicit ACL	Explicit ACL: Sequence	Explicit ACL: Default	Result
Allow (accept)	Deny (drop)	—	Deny (drop)
Allow (accept)	—	Deny (drop)	Allow (accept)
Deny (drop)	Allow (accept)	—	Allow (accept)
Deny (drop)	—	Allow (accept)	Deny (drop)

Additional Information

Configuring Localized Data Policy for IPv4
Localized Data Policy

Localized Data Policy Configuration Examples

This article provides some straightforward examples of configuring localized data policy to help you get an idea of how to use policy to influence traffic flow across the Viptela domain. Localized data policy, also known as access lists, is configured directly on the local vEdge routers.

QoS

You can configure quality of service (QoS) to classify data packets and control how traffic flows out of and in to the interfaces on a vEdge router and on the interface queues. For examples of how to configure a QoS policy, see **Forwarding and QoS Configuration Examples** .

Mirroring Example

This example illustrates how to configure a mirror instance to automatically send a copy of certain types of data packet to a specified destination for analysis. After you configure the mirror instance, include it in an access list. Here, "mirror-m1" is configured with the host at source address 10.20.23.16 and destination host at 10.2.2.11. The mirror instance is then included in the access list "acl2," which is configured so that data packets originating from the host at source address 10.20.24.17 and going to the destination host at 10.20.25.18 are mirrored to the destination host at 10.2.2.11 with the source address of the originating host as 10.20.23.16.

```
policy
 mirror m1
  remote-dest 10.2.2.11 source 10.20.23.16
 !
!

vEdge# show running-config policy access-list acl2
policy
 access-list acl2
  sequence 1
   match
    source-ip      10.20.24.17/32
    destination-ip 10.20.25.18/32
   !
   action accept
    mirror m1
   !
  !
  default-action drop
 !
!
```

Additional Information

Forwarding and QoS Configuration Examples

Policy Basics CLI Reference

CLI commands for configuring and monitoring policy.

Centralized Control Policy Command Hierarchy

Configure on vSmart controllers only.

```
policy
  lists
    color-list list-name
      color color
    prefix-list list-name
      ip-prefix prefix/length
    site-list list-name
```

```
        site-id site-id
      tloc-list list-name
        tloc address color color encap encapsulation [preference value weight value]
      vpn-list list-name
        vpn vpn-id
  policy
    control-policy policy-name
      default-action action
      sequence number
        match
          route
            color color
            color-list list-name
            omp-tag number
            origin protocol
            originator ip-address
            preference number
            prefix-list list-name
            site-id site-id
            site-list list-name
            tloc address
            tloc-list list-name
            vpn vpn-id
            vpn-list list-name
          tloc
            carrier carrier-name
            color color
            color-list list-name
            domain-id domain-id
            group-id group-id
            omp-tag number
            originator ip-address
            preference number
            site-id site-id
            site-list list-name
            tloc address
            tloc-list list-name
        action
          reject
          accept
            export-to (vpn vpn-id | vpn-list list-name)
            set
              omp-tag number
              preference value
              service service-name (tloc ip-address | tloc-list list-name) [vpn vpn-id]
              tloc-action action
              tloc-list list-name
  apply-policy
    site-list list-name control-policy policy-name (in | out)
```

Localized Control Policy Command Hierarchy

Configure on vEdge routers only.

```
policy
  lists
    as-path-list list-name
      as-path as-number
    community-list list-name
      community [aa:nn | internet | local-as | no-advertise | no-export]
    ext-community-list list-name
      community [rt (aa:nn | ip-address) | soo (aa:nn | ip-address)]
```

```
    prefix-list list-name
      ip-prefix prefix/length
policy
  route-policy policy-name
    default-action action
    sequence number
      match
        address list-name
        as-path list-name
        community list-name
        ext-community list-name
        local-preference number
        metric number
        next-hop list-name
        omp-tag number
        origin (egp | igp | incomplete)
        ospf-tag number
        peer address
      action
        reject
        accept
          set
            aggregator as-number ip-address
            as-path (exclude | prepend) as-number
            atomic-aggregate
            community value
            local-preference number
            metric number
            metric-type (type1 | type2)
            next-hop ip-address
            omp-tag number
            origin (egp | igp | incomplete)
            originator ip-address
            ospf-tag number
            weight number
vpn vpn-id
  router
    bgp local-as-number
      address-family ipv4_unicast
        redistribute (connected | nat | omp | ospf | static) [route-policy policy-name]
      neighbor address
        address-family ipv4-unicast
          route-policy policy-name (in | out)
    ospf
      redistribute (bgp | connected | nat | omp | static) route-policy policy-name
      route-policy policy-name in
```

Centralized Data Policy Command Hierarchy

Configure on vSmart controllers only.

```
policy
  lists
    app-list list-name
      (app applications | app-family application-families)
    data-prefix-list list-name
      ip-prefix prefix/length
    site-list list-name
      site-id site-id
    tloc-list list-name
      tloc ip-address color color encap encapsulation [preference value weight value]
```

```
      vpn-list list-name
        vpn vpn-id
  policy
    data-policy policy-name
      vpn-list list-name
        default-action action
        sequence number
          match
            app-list list-name
            destination-data-prefix-list list-name
            destination-ip prefix/length
            destination-port number
            dns (request | response)
            dns-app-list list-name
            dscp number
            packet-length number
            plp (high | low)
            protocol number
            source-data-prefix-list list-name
            source-ip prefix/length
            source-port number
            tcp flag
          action
            cflowd
            count counter-name
            drop
            log
            tcp-optimization
            accept
              nat [pool number] [use-vpn-0]
              redirect-dns (host | ip-address)
              set
                dscp number
                forwarding-class class
                local-tloc color color [encap encapsulation]
                local-tloc-list color color [encap encapsulation] [restrict]
                next-hop ip-address
                policer policer-name
                service service-name local [restrict] [vpn vpn-id]
                service service-name [tloc ip-address | tloc-list list-name] [vpn vpn-id]
                tloc ip-address color color [encap encapsulation]
                tloc-list list-name
                vpn vpn-id
    vpn-membership policy-name
      default-action action
      sequence number
        match
          vpn vpn-id
          vpn-list list-name
        action
          (accept | reject)
  apply-policy
    site-list list-name data-policy policy-name (all | from-service | from-tunnel)
    site-list list-name vpn-membership policy-name
```

Localized Data Policy Command Hierarchy

For IPv4

Configure on vEdge routers only.

```
policy
  lists
    prefix-list list-name
      ip-prefix prefix/length
  class-map
    class class-name queue number
  log-frequency number
  mirror mirror-name
    remote-dest ip-address source ip-address
  policer policer-name
    burst bytes
    exceed action
    rate bps
  qos-map map-name
    qos-scheduler scheduler-name
  qos-scheduler scheduler-name
    bandwidth-percent percentage
    buffer-percent percentage
    class class-name
    drops drop-type
  rewrite-rule rule-name
policy
  access-list acl-name
    default-action action
    sequence number
      match
        class class-name
        destination-data-prefix-list list-name
        destination-ip prefix/length
        destination-port number
        dscp number
        packet-length number
        plp (high | low)
        protocol number
        source-data-prefix-list list-name
        source-ip prefix-length
        source-port number
        tcp flag
      action
        drop
          count counter-name
          log
        accept
          class class-name
          count counter-name
          log
          mirror mirror-name
          policer policer-name
          set dscp value
vpn vpn-id
  interface interface-name
    access-list acl-name (in | out)
```

For IPv6

Configure on vEdge routers only.

```
policy ipv6
  class-map
    class class map map
  mirror mirror-name
```

```
      remote-dest ip-address source ip-address
    policer policer-name
      rate bandwidth
      burst bytes
      exceed action
  policy ipv6
    access-list list-name
      sequence number
        match
          match-parameters
        action
          drop
          count counter-name
          log
          accept
            class class-name
            mirror mirror-name
            policer policer-name
      default-action
        (accept | drop)
  vpn vpn-id
    interface interface-name
      ipv6 access-list list-name (in | out)
```

Operational Commands

show running-config

Additional Information

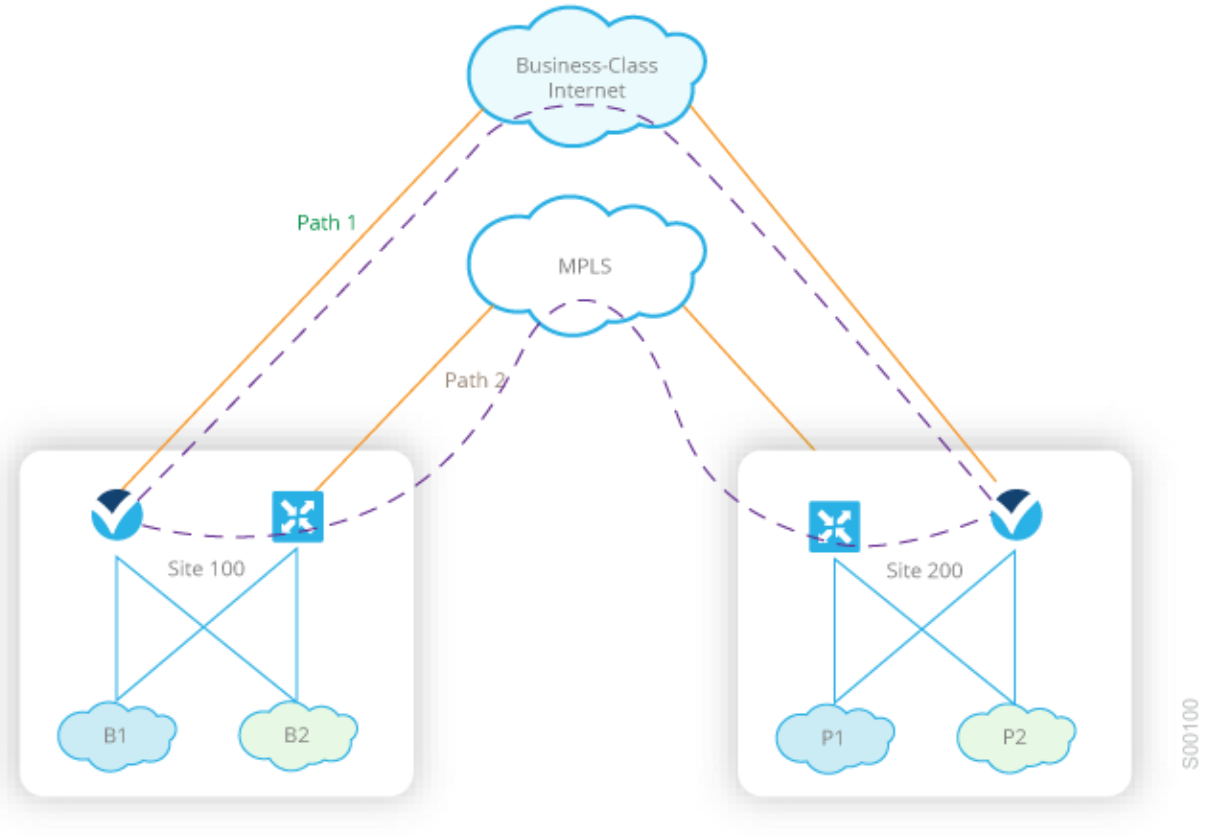
Centralized Control Policy
Centralized Data Policy
Localized Control Policy
Localized Data Policy
Policy Overview

Policy Applications

Application-Aware Routing

Application-aware routing tracks network and path characteristics of the data plane tunnels between vEdge routers and uses the collected information to compute optimal paths for data traffic. These characteristics include packet loss, latency, and jitter, and the load, cost and bandwidth of a link. The ability to consider factors in path selection other than those used by standard routing protocols—such as route prefixes, metrics, link-state information, and route removal on the edge router—offers a number of advantages to an enterprise:

- In normal network operation, the path taken by application data traffic through the network can be optimized, by directing it to WAN links that support the required levels of packet loss, latency, and jitter defined in an application's SLA.
- In the face of network soft failures, performance degradation can be minimized. The tracking of network and path conditions by application-aware routing in real time can quickly reveal performance issues, and it automatically activates strategies that redirect data traffic to the best available path. As the network recovers from the soft failure conditions, application-aware routing automatically readjusts the data traffic paths.
- Network costs can be reduced because data traffic can be more efficiently load-balanced.
- Application performance can be increased without the need for WAN upgrades.

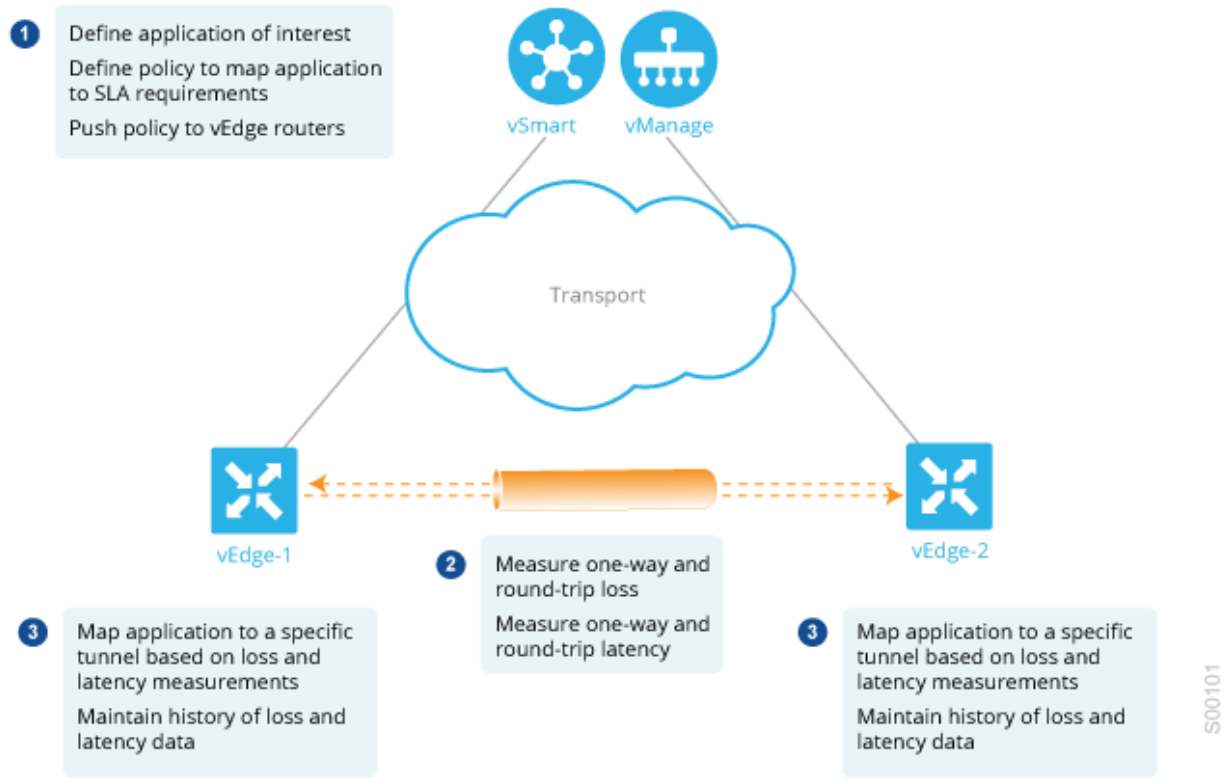


Each vEdge router supports up to eight TLOCs, allowing a single vEdge router to connect to up to eightdifferent WAN networks. This capability allows path customization for application traffic that has different needs in terms of packet loss and latency.

Components of Application-Aware Routing

The Viptela Application-Aware Routing solution consists of three elements:

- **Identification** —You define the application of interest, and then you create a centralized data policy that maps the application to specific SLA requirements. You single out data traffic of interest by matching on the Layer 3 and Layer 4 headers in the packets, including source and destination prefixes and ports, protocol, and DSCP field. As with all centralized data policies, you configure them on a vSmart controller, which then passes them to the appropriate vEdge routers.
- **Monitoring and measuring** —The Viptela software uses BFD packets to continuously monitor the data traffic on the data plane tunnels between vEdge routers, and periodically measures the performance characteristics of the tunnel. To gauge performance, the Viptela software looks for traffic loss on the tunnel, and it measures latency by looking at the one-way and round-trip times of traffic traveling over the tunnel. These measurements might indicate a failure or soft failure condition.
- **Mapping application traffic to a specific transport tunnel** —The final step is to map an application's data traffic to the data plane tunnel that provides the desired performance for the application. The mapping decision is based on two criteria: the best-path criteria computed from measurements performed on the WAN connections and on the constraints specified in a policy specific to application-aware routing.



To create data policy based on the Layer 7 application itself, use configure deep packet inspection with a centralized data policy. With deep packet inspection, you can direct traffic to a specific tunnel, based on the remote TLOC, the remote TLOC, or both. You cannot direct traffic to tunnels based on SLA classes.

Classification of Tunnels into SLA Classes

The process of classifying tunnels into one or more SLA classes for application-aware routing has three parts:

- Measure loss, latency, and jitter information for the tunnel.
- Calculate the average loss, latency, and jitter for the tunnel.

- Determine the SLA classification of the tunnel.

Measure Loss, Latency, and Jitter

When a data plane tunnel in the overlay network is established, a BFD session automatically starts on the tunnel. In the overlay network, each tunnel is identified with a color that identifies a specific link between a local TLOC and a remote TLOC. The BFD session monitors the liveness of the tunnel by periodically sending Hello packets to detect whether the link is operational. Application-aware routing uses the BFD Hello packets to measure the loss, latency, and jitter on the links.

By default, the BFD Hello packet interval is 1 second. This interval is user-configurable (with the **bfd color interval** command). Note that the BFD Hello packet interval is configurable per tunnel.

Calculate Average Loss, Latency, and Jitter

BFD periodically polls all the tunnels on the vEdge router to collect packet latency, loss, jitter, and other statistics for use by application-aware routing. At each poll interval, application-aware routing calculates the average loss, latency, and jitter for each tunnel, and then calculates or recalculates each tunnel's SLA. Each poll interval is also called a "bucket."

By default, the poll interval is 10 minutes. With the default BFD Hello packet interval at 1 second, this means that information from about 600 BFD Hello packets is used in one poll interval to calculate loss, latency, and jitter for the tunnel. The poll interval is user-configurable (with the **bfd app-route poll-interval** command). Note that the application-aware routing poll interval is configurable per vEdge router; that is, it applies to all tunnels originating on a router.

Reducing the poll interval without reducing the BFD Hello packet interval may affect the quality of the loss, latency, and jitter calculation. For example, setting the poll interval to 10 seconds when the BFD Hello packet interval is 1 second means that only 10 Hello packets are used to calculate the loss, latency, and jitter for the tunnel.

The loss, latency, and jitter information from each poll interval is preserved for six poll intervals. At the seventh poll interval, the information from the earliest polling interval is discarded to make way for the latest information. In this way, application-aware routing maintains a sliding window of tunnel loss, latency, and jitter information.

The number of poll intervals (6) is not user-configurable. Each poll interval is identified by an index number (0 through 5) in the output of the **show app-route statistics** command.

Determine the SLA Classification

To determine the SLA classification of a tunnel, application-aware routing uses the loss, latency, and jitter information from the latest poll intervals. The number of poll intervals used is determined by a multiplier. By default, the multiplier is 6, so the information from all the poll intervals (specifically, from the last six poll intervals) is used to determine the classification. For the default poll interval of 10 minutes and the default multiplier of 6, the loss, latency, and jitter information collected over the last hour is considered when classifying the SLA of each tunnel. These default values have to be chosen to provide damping of sorts, as a way to prevent frequent reclassification (flapping) of the tunnel.

The multiplier is user-configurable (with the **bfd app-route multiplier** command). Note that the application-aware routing multiplier is configurable per vEdge router; that is, it applies to all tunnels originating on a router.

If there is a need to react quickly to changes in tunnel characteristics, you can reduce the multiplier all the way down to 1. With a multiplier of 1, only the latest poll interval loss and latency values are used to determine whether this tunnel can satisfy one or more SLA criteria.

Based on the measurement and calculation of tunnel loss and latency, each tunnel may satisfy one or more user-configured SLA classes. For example, a tunnel with a mean loss of 0 packets and mean latency of 10 milliseconds would satisfy a class that has been defined with a

maximum packet loss of 5 and a minimum latency of 20 milliseconds, and it would also satisfy a class that has been defined with a maximum packet loss of 0 and minimum latency of 15 milliseconds.

Regardless of how quickly a tunnel is reclassified, the loss, latency, and jitter information is measured and calculated continuously. You can configure how quickly application-aware routing reacts to changes by modifying the poll interval and multiplier.

Additional Information

Application-Aware Routing Policy Configuration Example
Configuring Application-Aware Routing
Configuring Centralized Data Policy
Configuring Split DNS

Configuring Application-Aware Routing

This article provides general procedures for configuring application-aware routing.

Configuration Components

Application-aware routing consists of three components:

- Identify the applications of interest. To determine which applications are running on vEdge routers, you enable application visibility on these devices. Then you configure an application-aware routing policy on the vSmart controller, which defines the applications of interest and the data plane tunnel performance characteristics required to transmit an application's data traffic. These characteristics are called a *service-level agreement (SLA)* . The controller automatically pushes the policy to the appropriate vEdge routers.
- Monitor and measure data plane tunnel performance is done automatically and continuously by the vEdge routers, by tracking BFD Hello packets. Application-aware routing periodically polls the performance statistics to calculate the packet latency, loss, and jitter information for each tunnel. The default polling interval is good for most network situations, but you can modify it to meet specific business needs.
- Map application traffic to a specific data plane tunnel is done on the vEdge routers, based on the SLA requirements defined in application-aware routing policy and based on the real-time performance of the vEdge routers' data plane tunnels. You can modify how often a vEdge router calculates each tunnel's SLA and determines a tunnel's SLA classification.

Regardless of whether you have configured an application-aware routing policy, each vEdge router automatically monitors traffic jitter, loss, and latency and other interface characteristics on their data plane tunnels. To display this information, use the **show app-route stats** command on the vEdge router. You can use these statistics to determine the baseline performance characteristics of the vEdge router's tunnels. You can also use them to create appropriate application-aware routing policies and to modify existing policies.

Configure Application-Aware Routing Policy

Application-aware routing policy affects only traffic that is flowing from the service side (the local/WAN side) to the tunnel (WAN) side of the vEdge router.

An application-aware routing policy matches applications with an SLA, that is, with the data plane tunnel performance characteristics that are necessary to transmit the applications' data traffic. The primary purpose of application-aware routing policy is to optimize the path for data traffic being transmitted by vEdge routers.

An application-aware routing policy is a type of centralized data policy: you configure it on the vSmart controller, and the controller automatically pushes it to the affected vEdge routers. As with any policy, an application-aware routing policy consists of a series of numbered (ordered) sequences of match-action pairs that are evaluated in order, from lowest sequence number to highest sequence number. When a data packet matches one of the match conditions, an SLA action is applied to the packet to determine the data plane tunnel to use to transmit the packet. If a packet matches no parameters in any of the policy sequences, and if no default SLA class is configured, the packet is accepted and forwarded with no consideration of SLA. Because application-aware routing policy accepts

nonmatching traffic by default, it is considered to be a positive policy. Other types of policies in the Viptela software are negative policies, because by default they drop nonmatching traffic.

General Application-Aware Routing Policy vManage Configuration Procedure

To configure application-aware routing policy, use the vManage policy configuration wizard. The wizard consists of four sequential screens that guide you through the process of creating and editing policy components:

- Create Applications or Groups of Interest—Create lists that group together related items and that you call in the match or action components of a policy.
- Configure Topology—Create the network structure to which the policy applies.
- Configure Traffic Rules—Create the match and action conditions of a policy.
- Apply Policies to Sites and VPNs—Associate policy with sites and VPNs in the overlay network.

In the first three policy configuration wizard screens, you are creating policy components or blocks. In the last screen, you are applying policy blocks to sites and VPNs in the overlay network.

For a application-aware routing policy to take effect, you must activate the policy.

Start the Policy Configuration Wizard

To start the policy configuration wizard:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy.

The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed.

Create Applications or Groups of Interest

To create lists of applications or groups to use in centralized policy:

1. Start the policy configuration wizard as explained above.
2. Create new lists , as described in the following table:

List Type	Procedure

Application	1. In the left bar, click Application. 2. Click New Application List. 3. Enter a name for the list. 4. Click either the Application or Application Family button. 5. From the Select drop-down, select the desired applications or application families. 6. Click Add. Two application lists are preconfigured. You cannot edit or delete these lists. 7. Google_Apps—Includes Google applications, such as gmail, Google maps, and YouTube. To display a full list of Google applications, click the list in the Entries column. 8. Microsoft_Apps—Includes Microsoft applications, such as Excel, Skype, and Xbox. To display a full list of Microsoft applications, click the list in the Entries column.
Prefix	9. In the left bar, click Prefix. 10. Click New Prefix List. 11. Enter a name for the list. 12. In the Add Prefix field, enter one or more data prefixes separated by commas. 13. Click Add.
Site	14. In the left bar, click Site. 15. Click New Site List. 16. Enter a name for the list. 17. In the Add Site field, enter one or more site IDs separated by commas. 18. Click Add.
SLA Class	19. In the left bar, click SLA Class. 20. Click New SLA Class List. 21. Enter a name for the list. 22. Define the SLA class parameters: a. In the Loss field, enter the maximum packet loss on the connection, a value from 0 through 100 percent. b. In the Latency field, enter the maximum packet latency on the connection, a value from 0 through 1,000 milliseconds. c. In the Jitter field, enter the maximum jitter on the connection, a value from 1 through 1,000 milliseconds. 23. Click Add.

VPN	24. In the left bar, click VPN. 25. Click New VPN List. 26. Enter a name for the list. 27. In the Add VPN field, enter one or more VPN IDs separated by commas. 28. Click Add.

4. Click Next to move to Configure Topology in the wizard. When you first open this screen, the Topology tab is selected by default.

Configure the Network Topology

To configure the network topology or a VPN membership to use in centralized policy:

1. If you are already in the policy configuration wizard, skip to Step 4. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Create a network topology, as described in the following table:

Policy Type	Description	Procedure

Hub and Spoke	Policy for a topology with one or more central hub sites and with spokes connected to a hub	1. In the Add Topology drop-down, select Hub and Spoke. 2. Enter a name for the hub-and-spoke policy. 3. Enter a description for the policy. 4. In the VPN List field, select the VPN list for the policy. 5. In the left pane, click Add Hub and Spoke. A hub-and-spoke policy component containing the text string My Hub-and-Spoke is added in the left pane. 6. Double-click the My Hub-and-Spoke text string, and enter a name for the policy component. 7. In the right pane, add hub sites to the network topology: a. Click Add Hub Sites. b. In the Site List Field, select a site list for the policy component. c. Click Add. d. Repeat Steps 7a, 7b, and 7c to add more hub sites to the policy component. 8. In the right pane, add spoke sites to the network topology: a. Click Add Spoke Sites. b. In the Site List Field, select a site list for the policy component. c. Click Add. d. Repeat Steps 8a, 8b, and 8c to add more spoke sites to the policy component. 9. Repeat Steps 5 through 8 to add more components to the hub-and-spoke policy. 10. Click Save Hub and Spoke Policy.
Mesh	Partial-mesh or full-mesh region	11. In the Add Topology drop-down, select Mesh. 12. Enter a name for the mesh region policy component. 13. Enter a description for the mesh region policy component. 14. In the VPN List field, select the VPN list for the policy. 15. Click New Mesh Region. 16. In the Mesh Region Name field, enter a name for the individual mesh region. 17. In the Site List field, select one or more sites to include in the mesh region. 18. Repeat Steps 5 through 7 to add more mesh regions to the policy. 19. Click Save Mesh Region.

5. To use an existing topology:

   a. In the Add Topology drop-down, click Import Existing Topology. The Import Existing Topology popup displays.

b. Select the type of topology.

c. In the Policy drop-down, select the name of the topology.

d. Click Import.

6. Click Next to move to Configure Traffic Rules in the wizard. When you first open this screen, the Application-Aware Routing tab is selected by default.

Configure Traffic Rules

To create the match and action rules to apply to traffic affected by the policy:

1. If you are already in the policy configuration wizard, skip this procedure. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.

To configure traffic rules for application-aware routing policy:

1. In the Application-Aware Routing bar, select the Application-Aware Routing tab.
2. Click the Add Policy drop-down.
3. Select Create New, and in the left pane, click Sequence Type. A policy sequence containing the text string App Route is added in the left pane.
4. Double-click the App Route text string, and enter a name for the policy sequence.The name you type is displayed both in the Sequence Type list in the left pane and in the right pane.
5. In the right pane, click Sequence Rule. The Match/Action box opens, and Match is selected by default. The available policy match conditions are listed below the box.
6. To select one or more Match conditions, click its box and set the values as described in the following table:

Match Condition	Procedure
None (match all packets)	Do not specify any match conditions.
Applications/Application Family List	1. In the Match conditions, click Applications/Application Family List. 2. In the drop-down, select the application family. 3. To create an application list: a. Click New Application List. b. Enter a name for the list. c. Click the Application button to create a list of individual applications. Click the Application Family button to create a list of related applications. d. In the Select Application drop-down, select the desired applications or application families. e. Click Save.

Destination Data Prefix	4. In the Match conditions, click Destination Data Prefix. 5. To match a list of destination prefixes, select the list from the drop-down. 6. To match an individual destination prefix, type the prefix in the Destination box.
Destination Port	7. In the Match conditions, click Destination Port. 8. In the Destination field, enter the port number. Specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-]).
DNS Application List (to enable split DNS)	9. In the Match conditions, click DNS Application List. 10. In the drop-down, select the application family.
DNS (to enable split DNS)	11. In the Match conditions, click DNS. 12. In the drop-down, select Request to process DNS requests for the DNS applications, and select Response to process DNS responses for the applications.
DSCP	13. In the Match conditions, click DSCP. 14. In the DSCP field, type the DSCP value, a number from 0 through 63.
PLP	15. In the Match conditions, click PLP. 16. In the PLP drop-down, select Low or High. To set the PLP to high, apply a policer that includes the **exceed remark** option.
Protocol	17. In the Match conditions, click Protocol. 18. In the Protocol field, type the Internet Protocol number, a number from 0 through 255.
Source Data Prefix	19. In the Match conditions, click Source Data Prefix. 20. To match a list of source prefixes, select the list from the drop-down. 21. To match an individual source prefix, type the prefix in the Source box.
Source Port	22. In the Match conditions, click Source Port. 23. In the Source field, enter the port number. Specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-]).

7. To select actions to take on matching data traffic, click the Actions box. The available policy actions are listed below the box.

8. Set the policy action as described in the following table:

Match Condition	Description	Procedure
Backup SLA Preferred Color	When no tunnel matches the SLA, direct the data traffic to a specific tunnel. Data traffic is sent out the configured tunnel if that tunnel interface is available. If that tunnel interface is not available, traffic is sent out another available tunnel. You can specify one or more colors. The backup SLA preferred color is a loose matching, not a strict matching.	1. In the Action conditions, click Backup SLA Preferred Color. 2. In the drop-down, select one or more colors.

Counter	Count matching data packets.	3. In the Action conditions, click Counter. 4. In the Counter Name field, enter the name of the file in which to store packet counters.
Log	Place a sampled set of packets that match the SLA class rule into system logging (syslog) files. In addition to logging the packet headers, a syslog message is generated the first time a packet header is logged and then every 5 minutes thereafter, as long as the flow is active.	5. In the Action conditions, click Log to enable logging.
SLA Class List	For the SLA class, all matching data traffic is directed to a tunnel whose performance matches the SLA parameters defined in the class. The software first tries to send the traffic through a tunnel that matches the SLA. If a single tunnel matches the SLA, data traffic is sent through that tunnel. If two or more tunnels match, traffic is distributed among them. If no tunnel matches the SLA, data traffic is sent through one of the available tunnels.	6. In the Action conditions, click SLA Class List. 7. In the SLA Class drop-down, select one or more SLA classes. 8. Optionally, in the Preferred Color drop-down, select the color of the data plane tunnel or tunnels to prefer. Traffic is load-balanced across all tunnels. If no tunnels match the SLA, data traffic is sent through any available tunnel. That is, color preference is a loose matching, not a strict matching. 9. Click Strict to perform strict matching of the SLA class. If no data plane tunnel is available that satisfies the SLA criteria, traffic is dropped.

9. Click Save Match and Actions.
10. Create additional sequence rules as desired. Drag and drop to re-arrange them.
11. Create additional sequence types as desired. Drag and drop to re-arrange them.
12. Click Save Application-Aware Routing Policy.

Click Next to move to Apply Policies to Sites and VPNs in the wizard.

Apply Policies to Sites and VPNs

In the last screen of the policy configuration wizard, you associate the policy blocks that you created on the previous three screens with VPNs and with sites in the overlay network.

To apply a policy block to sites and VPNs in the overlay network:

1. If you are already in the policy configuration wizard, skip to Step 6. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.
5. Click Next. The Apply Policies to Sites and VPNs screen opens.
6. In the Policy Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.

7. In the Policy Description field, enter a description of the policy. It can contain up to 2048 characters. This field is mandatory, and it can contain any characters and spaces.
8. From the Topology bar, select the type of policy block. The table then lists policies that you have created for that type of policy block.
9. Click Add New Site List and VPN list. Select one or more site lists, and select one or more VPN lists. Click Add.
10. Click Preview to view the configured policy. The policy is displayed in CLI format.
11. Click Save Policy. The Configuration ► Policies screen opens, and the policies table includes the newly created policy.

Activate an Application-Aware Routing Policy

Activating an application-aware routing policy sends that policy to all connected vSmart controllers. To activate a policy:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Select a policy.
3. Click the More Actions icon to the right of the row, and click Activate. The Activate Policy popup opens. It lists the IP addresses of the reachable vSmart controllers to which the policy is to be applied.
4. Click Activate.

General Application-Aware Routing Policy CLI Configuration Procedure

Following are the high-level steps for configuring an application-aware routing policy:

1. Create a list of overlay network sties to which the application-aware routing policy is to be applied (in the **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-site-list)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–). Create additional site lists, as needed.
2. Create SLA classes and traffic characteristics to apply to matching application data traffic:
   vSmart(config)# **policy sla-class** *sla-class-name*
   vSmart(config-sla-class)# **jitter** *milliseconds*
   vSmart(config-sla-class)# **latency** *milliseconds*
   vSmart(config-sla-class)# **loss** *percentage*
3. Create lists of applications, IP prefixes, and VPNs to use in identifying application traffic of interest (in the **match** section of the policy definition):
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **app-list** *list-name*
   vSmart(config-app-list)# (**app** *application-name* | **app-family** *family-name*)

   vSmart(config-lists)# **prefix-list** *list-name*
   vSmart(config-prefix-list)# **ip-prefix** *prefix / length*

   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-vpn-list)# **vpn** *vpn-id*
4. If you are configuring a logging action, configure how often to log packets to syslog files:
   vEdge(config)# **policy log-frequency** *number*

5. Create an application-aware routing policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy app-route-policy** *policy-name*
   vSmart(config-app-route-policy)# **vpn-list** *list-name*

6. Within the policy, create one or more numbered sequence of match–action pairs, where the match parameters define the data traffic and applications of interest and the action parameters specify the SLA class to apply if a match occurs.

   a. Create a sequence:
      vSmart(config-app-route-policy)# **sequence** *number*

   b. Define match parameters for data packets:
      vSmart(config-sequence)# **match** *parameters*

   c. Define the action to take if a match occurs:
      vSmart(config-sequence)# **action sla-class** *sla-class-name* [**strict**]
      vSmart(config-sequence)# **action sla-class** *sla-class-name* [**strict**] **preferred-color** *colors*
      vSmart(config-sequence)# **action backup-sla-preferred-color** *colors*
      The first two **action** options direct matching data traffic to a tunnel interface that meets the SLA characteristics in the specified SLA class:
      • **sla-class** *sla-class-name* —When you specify an SLA class with no additional parameters, data traffic that matches the SLA is forwarded as long as one tunnel interface is available. The software first tries to send the traffic through a tunnel that matches the SLA. If a single tunnel matches the SLA, data traffic is sent through that tunnel. If two or more tunnels match, traffic is distributed among them. If no tunnel matches the SLA, data traffic is sent through one of the available tunnels.
      • **sla-class** *sla-class-name* **preferred-color** *color* —To set a specific tunnel to use when data traffic matches an SLA class, include the **preferred-color** option, specifying the color of the preferred tunnel. If more than one tunnel matches the SLA, traffic is sent to the preferred tunnel. If a tunnel of the preferred color is not available, traffic is sent through any tunnel that matches the SLA class. If no tunnel matches the SLA, data traffic is sent through any available tunnel. In this sense, color preference is considered to be a loose matching, not a strict matching, because data traffic is always forwarded, whether a tunnel of the preferred color is available or not.
      • **sla-class** *sla-class-name* **preferred-color** *colors* —To set multiple tunnels to use when data traffic matches an SLA class, include the **preferred-color** option, specifying two or more tunnel colors. Traffic is load-balanced across all tunnels. If no tunnel matches the SLA, data traffic is sent through any available tunnel. In this sense, color preference is considered to be a loose matching, not a strict matching, because data traffic is always forwarded, whether a tunnel of the preferred color is available or not.
      When no tunnel matches the SLA, you can choose how to handle the data traffic:
      • **strict** —Drop the data traffic.
      • **backup-sla-preferred-color** *colors* —Direct the data traffic to a specific tunnel. Data traffic is sent out the configured tunnel if that tunnel interface is available; if that tunnel is unavailable, traffic is sent out another available tunnel. You can specify one or more colors. As with the **preferred-color** option, the backup SLA preferred color is loose matching.
      In a single **action** configuration, you cannot include both the **strict** and **backup-sla-preferred-color** options.

   d. Count the packets or bytes that match the policy:
      vSmart(config-sequence)# **action count** *counter-name*

   e. Place a sampled set of packets that match the SLA class rull into syslog files:
      vSmart(config-sequence)# **action log**

   f. The match–action pairs within a policy are evaluated in numerical order, based on the sequence number, starting with the lowest number. If a match occurs, the corresponding action is taken and policy evaluation stops.

7. If a packet does not match any of the conditions in one of the sequences, a default action is taken. For application-aware routing policy, the default action is to accept nonmatching traffic and forward it with no consideration of SLA. You can configure the default action so that SLA parameters are applied to nonmatching packets:
   vSmart(config- *policy-name*)# **default-action sla-class** *sla-class-name*

8. Apply the policy to a site list:
   vSmart(config)# **apply-policy site-list** *list-name* **app-route-policy** *policy-name*

Structural Components of Policy Configuration for Application-Aware Routing

Here are the structural components required to configure application-aware routing policy. Each one is explained in more detail in the sections below.

```
policy
  lists
    app-list list-name
      (app application-name | app-family application-family)
    prefix-list list-name
      ip-prefix prefix
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn-id vpn-id
 log-frequency number
 sla-class sla-class-name
   jitter milliseconds
   latency milliseconds
   loss percentage
 app-route-policy policy-name
   vpn-list list-name
     sequence number
       match
         match-parameters
       action
         backup-sla-preferred-color colors
         count counter-name
         log
         sla-class sla-class-name [strict] [preferred-color colors]
     default-action
       sla-class sla-class-name
apply-policy site-list list-name
  app-route-policy policy-name
```

Lists

Application-aware routing policy uses the following types of lists to group related items. You configure these lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command
Applications and application families	List of one or more applications or application families running on the subnets connected to the vEdge router. Each **app-list** can contain either applications or application families, but you cannot mix the two. To configure multiple applications or application families in a single list, include multiple **app** or **app-family** options, specifying one application or application family in each **app** or **app-family** option. • *application-name* is the name of an application. The Viptela software supports about 2300 different applications. To list the supported applications, use the ? in the CLI. • *application-family* is the name of an application family. It can one of the following: **antivirus** , **application-service** , **audio_video** , **authentication** , **behavioral** , **compression** , **database** , **encrypted** , **erp** , **file-server** , **file-transfer** , **forum** , **game** , **instant-messaging** , **mail** , **microsoft-office** , **middleware** , **network-management** , **network-service** , **peer-to-peer** , **printer** , **routing** , **security-service** , **standard** , **telephony** , **terminal** , **thin-client** , **tunneling** , **wap** , **web** , and **webmail** .	**app-list** *list-name* (**app** *application-name* | **app-family** *application-family*)
Data prefixes	List of one or more IP prefixes. To configure multiple prefixes in a single list, include multiple **ip-prefix** options, specifying one prefix in each option.	**data-prefix-list** *list-name* **ip-prefix** *prefix / length*

Sites	List of one or more site identifiers in the overlay network. To configure multiple sites in a single list, include multiple **site-id** options, specifying one site number in each option. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*
VPNs	List of one or more VPNs in the overlay network. To configure multiple VPNs in a single list, include multiple **vpn** options, specifying one VPN number in each option. You can specify a single VPN identifier (such as **vpn-id 1**) or a range of VPN identifiers (such as **vpn-id 1-10**).	**vpn-list** *list-name* **vpn** *vpn-id*

In the vSmart controller configuration, you can create multiple iterations of each type of list. For example, it is common to create multiple site lists and multiple VPN lists so that you can apply data policy to different sites and different customer VPNs across the network.

When you create multiple iterations of a type of list (for example, when you create multiple VPN lists), you can include the same values or overlapping values in more than one of these list. You can do this either on purpose, to meet the design needs of your network, or you can do this accidentally, which might occur when you use ranges to specify values. (You can use ranges to specify data prefixes, site identifiers, and VPNs.) Here are two examples of lists that are configured with ranges and that contain overlapping values:

- **vpn-list list-1 vpn 1-10**
  vpn-list list-2 vpn 6-8
- **site-list list-1 site 1-10**
  site-list list-2 site 5-15

When you configure data policies that contain lists with overlapping values, or when you apply data policies, you must ensure that the lists included in the policies, or included when applying the policies, do not contain overlapping values. To do this, you must manually audit your configurations. The Viptela configuration software performs no validation on the contents of lists, on the data policies themselves, or on how the policies are applied to ensure that there are no overlapping values.

If you configure or apply data policies that contain lists with overlapping values to the same site, one policy is applied and the others are ignored. Which policy is applied is a function of the internal behavior of Viptela software when it processes the configuration. This decision is not under user control, so the outcome is not predictable.

Logging Frequency

If you configure a logging action, by default, the vEdge router logs all data packet headers to a syslog file. To log only a sample of the data packet headers:

vEdge(config)# **policy log-frequency** *number*

number specifies how often to to log packet headers. For example, if you configure **log-frequency 20** , every sixteenth packet is logged. While you can configure any integer value for the frequency, the software rounds the value down to the nearest power of 2.

SLA Classes

The action taken in application-aware routing is applied based on what is called an SLA (a service-level agreement). An SLA class is defined by the maximum jitter, maximum latency, maximum packet loss, or a combination of these values, for the vEdge router's data plane tunnels. (Each tunnel is defined by a local TLOC–remote TLOC pair.) You configure SLA classes under the **policy sla-class** command hierarchy on vSmart controllers. You can configure a maximum of four SLA classes.

You can configure the following parameters in an SLA class:

Description	Command	Value or Range
Maximum acceptable packet jitter on the data plane tunnel	**jitter** *milliseconds*	1 through 1000 milliseconds
Maximum acceptable packet latency on the data plane tunnel.	**latency** *milliseconds*	1 through 1000 milliseconds

Maximum acceptable packet loss on the data plane tunnel.	**loss** *percentage*	0 through 100 percent

VPN Lists

Each application-aware policy instance is associated with a VPN list. You configure VPN lists with the **policy app-route-policy vpn-list** command. The VPN list you specify must be one that you created with a **policy lists vpn-list** command.

Sequences

Within each VPN list, an application-aware policy contains sequences of match–action pairs. The sequences are numbered to set the order in which data traffic is analyzed by the match–action pairs in the policy. You configure sequences with the **policy app-aware-policy vpn-list sequence** command.

Each sequence in an application-aware policy can contain one **match** command and one **action** command.

Match Parameters

Application-aware routing policy can match IP prefixes and fields in the IP headers. You configure the match parameters with the **match** command under the **policy app-route-policy vpn-list sequence** command hierarchy on vSmart controllers.

You can match these parameters:

Description	Command	Value or Range
Match all packets	Omit **match** command	—
Applications or application families	**app-list** *list-name*	Name of an **app-list** list
Group of destination prefixes	**destination-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual destination prefix	**destination-ip** *prefix / length*	IP prefix and prefix length
Destination port number	**destination-port** *number*	0 through 65535. Specify a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-]).
DSCP value	**dscp** *number*	0 through 63
Internet Protocol number	**protocol** *number*	0 through 255
Packet loss priority (PLP)	**plp**	(**high** | **low**) By default, packets have a PLP value of **low** . To set the PLP value to **high** , apply a policer that includes the **exceed remark** option.
Group of source prefixes	**source-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual source prefix	**source-ip** *prefix / length*	IP prefix and prefix length

Source port number	**source-port** *number*	0 through 65535; enter a single port number, a list of port numbers (with numbers separated by a space), or a range of port numbers (with the two numbers separated with a hyphen [-])
Split DNS, to resolve and process DNS requests on an application-by-application basis	**dns-app-list** *list-name* **dns** (**request** | **response**)	Name of an **app-list** list. This list specifies the applications whose DNS requests are processed. To process DNS requests sent by the applications (for outbound DNS queries), specify **dns request** . To process DNS responses returned from DNS servers to the applications, specify **dns response** .

Action Parameters

When data traffic matches the match parameters, the specified action is applied to it. For application-aware routing policy, the action is to apply an SLA class. The SLA class defines the maximum packet latency or maximum packet loss, or both, that the application allows on the data plane tunnel used to transmit its data. The Viptela software examines the recently measured performance characteristics of the data plane tunnels and directs the data traffic to the WAN connection that meets the specified SLA.

The following actions can be configured:

Description	Command	Value or Range
When no tunnel matches the SLA, direct the data traffic to a specific tunnel. Data traffic is sent out the configured tunnel if that tunnel interface is available. If that tunnel is unavailable, traffic is sent out another available tunnel. You can specify one or more colors. The backup SLA preferred color is a loose matching, not a strict matching.	**backup-sla-preferred-color** *colors*	**3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green** **lte** , **metro-ethernet** , **mpls** , **private1** through **private6** , **public-internet** , **red** , **silver**
Count matching data packets.	**action count** *counter-name*	Name of a counter.
Place a sampled set of packets that match the SLA class rule into the messages and vsyslog system logging (syslog) files. In addition to logging the packet headers, a syslog message is generated the first time a packet header is logged and then every 5 minutes thereafter, as long as the flow is active.	**action log**	To display logging information, use the **show app log flow-all** , **show app log flows** , and **show log** commands on the vEdge router.
SLA class to match. All matching data traffic is directed to a tunnel whose performance matches the SLA parameters defined in the class. The software first tries to send the traffic through a tunnel that matches the SLA. If a single tunnel matches the SLA, data traffic is sent through that tunnel. If two or more tunnels match, traffic is distributed among them. If no tunnel matches the SLA, data traffic is sent through one of the available tunnels.	**action sla-class** *sla-class-name*	SLA class name defined in **policy sla-class** command
Group of data plane tunnel colors to prefer when an SLA class match occurs. Traffic is load-balanced across all tunnels. If no tunnels match the SLA, data traffic is sent through any available tunnel. That is, color preference is a loose matching, not a strict matching.	**action sla-class** *sla-class-name* **preferred-color** *colors*	SLA class name defined in **policy sla-class** command and one of the supported tunnel colors.

Strict matching of the SLA class. If no data plane tunnel is available that satisfies the SLA criteria, traffic is dropped. Note that for policy configured with this option, data traffic that matches the match conditions is dropped until the application-aware routing path is established.	**action sla-class** *sla-class-name* **strict** **action sla-class** *sla-class-name* **preferred-color** *color* **strict** **action sla-class** *sla-class-name* **preferred-color** *colors* **strict**	SLA class name defined in **policy sla-class** command

If more than one data plane tunnel satisfies an SLA class criteria, the vEdge router selects one of them by performing load-balancing across the equal paths.

Default Action

A policy's default action defines how to handle packets that match none of the match conditions. For application-aware routing policy, if you do not configure a default action, all data packets are accepted and transmitted based on normal routing decisions, with no consideration of SLA.

To modify this behavior, include the **default-action sla-class** *sla-class-name* command in the policy, specifying the name of an SLA class you defined in the **policy sla-class** command.

When you apply an SLA class in a policy's default action, you cannot specify the **strict** option.

If no data plane tunnel satisfies the SLA class in the default action, the vEdge router selects one of the available tunnels by performing load-balancing across equal paths.

Applying Application-Aware Routing Policy

For an application-aware route policy to take effect, you apply it to a list of sites in the overlay network:

```
vSmart(config)# apply-policy site-list list-name app-route-policy policy-name
```

When you apply the policy, you do not specify a direction (either inbound or outbound). Application-aware routing policy affects only the outbound traffic on the vEdge routers.

For all **app-route-policy** policies that you apply with **apply-policy** commands, the site IDs across all the site lists must be unique. That is, the site lists must not contain overlapping site IDs. An example of overlapping site IDs are those in the two site lists **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** . Here, sites 70 through 100 are in both lists. If you were to apply these two site lists to two different **app-route-policy** policies, the attempt to commit the configuration on the vSmart controller would fail.

The same type of restriction also applies to the following types of policies:

- Centralized control policy (**control-policy**)
- Centralized data policy (**data-policy**)
- Centralized data policy used for cflowd flow monitoring (**data-policy** hat includes a **cflowd** action and **apply-policy** that includes a **cflowd-template** command)

You can, however, have overlapping site IDs for site lists that you apply for different types of policy. For example, the sites lists for **app-route-policy** and **data-policy** policies can have overlapping site IDs. So for the two example site lists above, **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** , you could apply one to a control policy and the other to a data policy.

As soon as you successfully activate the configuration on the vSmart controller by issuing a **commit** command, the controller pushes the application-aware routing policy to the vEdge routers at the specified sites.

To view the policy configured on the vSmart controller, use the **show running-config** command on the controller.

To view the policy that the vSmart controller has pushed to the vEdge router, issue the **show policy from-vsmart** command on the router.

To display flow information for the application-aware applications running on the vEdge router, issue the **show app dpi flows** command on the router.

How Application-Aware Routing Policy Is Applied in Combination with Other Data Policies

If you configure a vEdge router with application-aware routing policy and with other policies, the policies are applied to data traffic sequentially.

On a vEdge router, you can configure the following types of data policy:

- Centralized data policy. You configure this policy on the vSmart controller, and the policy is passed to the vEdge router. You define the configuration with the **policy data-policy configuration** command, and you apply it with the **apply-policy site-list data-policy** or **apply-policy site-list vpn-membership** command.
- Localized data policy, which is commonly called access lists. You configure access lists on the vEdge router with the **policy access-list** configuration command. You apply them, within a VPN, to an incoming interface with the **vpn interface access-list in** configuration command or to an outgoing interface with the **vpn interface access-list out** command.
- Application-aware routing policy. Application-aware routing policy affects only traffic that is flowing from the service side (the local/LAN side) to the tunnel (WAN) side of the vEdge router. You configure application-aware routing policy on the vSmart controller with the **policy app-route-policy** configuration command, and you apply it with the **apply-policy site-list app-route-policy** command. When you commit the configuration, the policy is passed to the appropriate vEdge routers. Then, matching data traffic on the vEdge routers is processed in accordance with the configured SLA conditions. Any data traffic that is not dropped as a result of this policy is passed to the data policy for evaluation. If the data traffic does not match and if no default action is configured, transmit it without SLA consideration.

You can apply only one data policy and one application-aware routing policy to a single site in the overlay network. When you define and apply multiple site lists in a configuration, you must ensure that a single data policy or a single application-aware routing policy is not applied to more than one site. The CLI does not check for this circumstance, and the **validate** configuration command does not detect whether multiple policies of the same type are applied to a single site.

For data traffic flowing from the service side of the router to the WAN side of the router, policy evaluation of the traffic evaluation occurs in the following order:

1. Apply the input access list on the LAN interface. Any data traffic that is not dropped as a result of this access list is passed to the application-aware routing policy for evaluation.
2. Apply the application-aware routing policy. Any data traffic that is not dropped as a result of this policy is passed to the data policy for evaluation. If the data traffic does not match and if no default action is configured, transmit it without SLA consideration.
3. Apply the centralized data policy. Any data traffic that is not dropped as a result of the input access list is passed to the output access list for evaluation.
4. Apply the output access list on the WAN interface. Any data traffic that is not dropped as a result of the output access list is transmitted out the WAN interface.

For data traffic coming from the WAN through the router and into the service-side LAN, the policy evaluation of the traffic occurs in the following order:

1. Apply the input access list on the WAN interface. Any data traffic that is not dropped as a result of the input access list is passed to the data policy for evaluation.

2. Apply the data policy. Any data traffic that is not dropped as a result of the input access list is passed to the output access list for evaluation.

3. Apply the output access list on the LAN interface. Any data traffic that is not dropped as a result of the output access list is transmitted out the LAN interface, towards its destination at the local site.

As mentioned above, application-aware routing policy affects only traffic that is flowing from the service side (the local/LAN side) to the tunnel (WAN) side of the vEdge router, so data traffic inbound from the WAN is processed only by access lists and data policy.

Configure the Monitoring of Data Plane Tunnel Performance

The Bidirectional Forwarding Detection (BFD) protocol runs over all data plane tunnels between vEdge routers, monitoring the liveness, and network and path characteristics of the tunnels. Application-aware routing uses the information gathered by BFD to determine the transmission performance of the tunnels. Performance is reported in terms of packet latency and packet loss on the tunnel.

BFD sends Hello packets periodically to test the liveness of a data plane tunnel and to check for faults on the tunnel. These Hello packets provide a measurement of packet loss and packet latency on the tunnel. The vEdge router records the packet loss and latency statistics over a sliding window of time. BFD keeps track of the six most recent sliding windows of statistics, placing each set of statistics in a separate bucket. If you configure an application-aware routing policy for the vEdge router, it is these statistics that the router uses to determine whether a data plane tunnel's performance matches the requirements of the policy's SLA.

The following parameters determine the size of the sliding window:

Parameters	Default	Configuration Command	Range
BFD Hello packet interval	1 second	**bfd color** *color* **hello-interval** *seconds*	1 through 65535 seconds
Polling interval for application-aware routing	10 minutes (600,000 milliseconds)	**bfd app-route poll-interval** *milliseconds*	1 through 4,294,967 ($2^{32} - 1$) milliseconds
Multiplier for application-aware routing	6	**bfd app-route multiplier** *number*	1 through 6

Let's use the default values for these parameters to explain how application-aware routing works:

- For each sliding window time period, application-aware routing sees 600 BFD Hello packets (BFD Hello interval x polling interval: 1 second x 600 seconds = 600 Hello packets). These packets provide measurements of packet loss and latency on the data plane tunnels.
- Application-aware routing retains the statistics for 1 hour (polling interval x multiplier: 10 minutes x 6 = 60 minutes).
- The statistics are placed in six separate buckets, indexed with the numbers 0 through 5. Bucket 0 contains the latest statistics, and bucket 5 the oldest. Every 10 minutes, the newest statistics are placed in bucket 0, the statistics in bucket 5 are discarded, and the remaining statistics move into the next bucket.
- Every 60 minutes (every hour), application-aware routing acts on the loss and latency statistics. It calculates the mean of the loss and latency of all the buckets in all the sliding windows and compares this value to the specified SLAs for the tunnel. If the calculated value satisfies the SLA, application-aware routing does nothing. If the value does not satisfy the SLA, application-aware routing calculates a new tunnel.
- Application-aware routing uses the values in all six buckets to calculate the mean loss and latency for a data tunnel. This is because the multiplier is 6. While application-aware always retains six buckets of data, the multiplier determines how many it actually uses to calculate the loss and latency. For example, if the multiplier is 3, buckets 0, 1, and 2 are used.

Because these default values take action only every hour, they work well for a stable network. To capture network failures more quickly so that application-aware routing can calculate new tunnels more often, adjust the values of these three parameters. For example, if you change just the polling interval to 1 minute (60,000 milliseconds), application-aware routing reviews the tunnel performance characteristics every minute, but it performs its loss and latency calculations based on only 60 Hello packets. It may take more than 1 minute for application-aware routing to reset the tunnel if it calculates that a new tunnel is needed.

To display statistics for each data plane tunnel, use the **show app-route stats** command:

```
vEdge# show app-route stats


TX    RX
                                      SRC    DST    MEAN  MEAN            TOTAL          AVERAGE  AVERAGE
DATA  DATA
SRC IP        DST IP          PROTO   PORT   PORT   LOSS  LATENCY  INDEX  PACKETS  LOSS  LATENCY  JITTER
PKTS  PKTS
-------------------------------------------------------------------------------------------------------
------------
192.168.1.3   24.6.101.120    ipsec   12346  12346  0     22       0      596      0     21       2
0     0
                                                                  1      596      0     21       2
0     0
                                                                  2      596      0     21       2
0     0
                                                                  3      597      1     21       2
0     0
                                                                  4      596      0     21       2
0     0
                                                                  5      596      0     29       4
0     0
192.168.1.3   24.130.213.18   ipsec   12346  12346  0     24       0      596      0     24       3
0     0
                                                                  1      596      0     25       3
0     0
                                                                  2      596      0     25       3
0     0
                                                                  3      596      0     24       3
0     0
                                                                  4      596      0     24       3
0     0
                                                                  5      596      0     24       3
0     0
192.168.1.3   24.130.233.111  ipsec   12346  34083  0     21       0      596      0     21       3
0     0
                                                                  1      596      0     22       3
0     0
                                                                  2      596      0     22       3
0     0
                                                                  3      596      0     21       3
0     0
                                                                  4      596      0     21       3
0     0
                                                                  5      596      0     21       3
0     0
192.168.1.3   24.130.233.111  ipsec   12346  36464  0     23       0      596      0     23       3
0     0
                                                                  1      596      0     23       3
0     0
                                                                  2      596      0     24       3
0     0
                                                                  3      596      0     23       4
0     0
                                                                  4      596      0     23       4
0     0
                                                                  5      596      0     23       4
0     0
...
```

To display the next-hop information for an IP packet that a vEdge router sends out a service side interface, use the **show policy service-path** command. To view the similar information for packets that the router sends out a WAN transport tunnel interface, use the **show policy tunnel-path** command.

Enable Application Visibility on vEdge Routers

You can enable application visibility directly on vEdge routers, without configuring application-aware routing policy so that you can monitor all the applications running in all VPNs in the LAN. To do this, configure application visiblity on the router:

```
vEdge(config)# policy app-visibility
```

To monitor the applications, use the **show app dpi applications** and **show app dpi supported-applications** commands on the vEdge router.

Additional Information

Application-Aware Routing
Application-Aware Routing Policy Configuration Example
Configuring Centralized Data Policy
Configuring Split DNS

Application-Aware Routing Policy Configuration Example

This article shows a straightforward example of configuring application-aware routing policy. This example defines a policy that applies to ICMP traffic, directing it to links with latency of 50 milliseconds or less when such links are available.

Configuration Steps

You configure application-aware routing policy on a vSmart controller. The configuration consists of the following high-level components:

- Definition of the application (or applications)
- Definition of SLA parameters
- Definition of sites, prefixes, and VPNs
- Application-aware routing policy itself
- Specification of overlay network sites to which the policy is applied

The order in which you configure these components is immaterial from the point of view of the CLI. However, from an architectural design point of view, a logical order is to first define all the parameters that are invoked in the application-aware routing policy itself or that are used to apply the policy to various sites in the overlay network. Then, you specify the application-aware routing policy itself and the network sites to which you want to apply the policy.

Here is the procedure for configuring this application-aware routing policy on vSmart, which is a vSmart controller:

1. Define the SLA parameters to apply to matching ICMP traffic. In our example, we want to direct ICMP traffic to links that have a latency of 50 milliseconds or less:
   vSmart# **config**
   vSmart(config)# **policy sla-class test_sla_class latency 50**
   vSmart(config-sla-class-test_sla_class)#

2. Define the site and VPN lists to which we want to apply the application-aware routing policy:
   vSmart(config-sla-class-test_sla_class)# **exit**
   vSmart(config-sla-class-test_sla_class)# **lists vpn-list vpn_1_list vpn 1**
   vSmart(config-vpn-list-vpn_1_list)# **exit**
   vSmart(config-lists)# **site-list site_500 site-id 500**
   vSmart(config-site-list-site_500)#

3. Configure the application-aware routing policy. Note that in this example, we apply the policy to the application in two different ways: In sequences 1, 2, and 3, we specify the protocol number (protocol 1 is ICMP, protocol 6 is TCP, and protocol 17 is UDP).

```
vSmart(config-site-list-site_500)# exit
vSmart(config-lists)# exit
vSmart(config-policy)# app-route-policy test_app_route_policy
vSmart(config-app-route-policy-test_app_route_policy)# vpn-list vpn_1_list
vSmart(config-vpn-list-vpn_1_list)# sequence 1 match protocol 6
vSmart(config-match)# exit
vSmart(config-sequence-1)# action sla-class test_sla_class strict
vSmart(config-sequence-1)# exit
vSmart(config-vpn-list-vpn_1_list)# sequence 2 match protocol 17
vSmart(config-match)# exit
vSmart(config-sequence-2)# action sla-class test_sla_class
vSmart(config-sequence-2)# exit
vSmart(config-vpn-list-vpn_1_list)# sequence 3 match protocol 1
vSmart(config-match)# exit
vSmart(config-sequence-3)# action sla-class test_sla_class strict
vSmart(config-sequence-3)# exit
vSmart(config-sequence-4)#
```

4. Apply the policy to the desired sites in the Viptela overlay network:

```
vSmart(config-sequence-4)# top
vSmart(config)# apply-policy site-list site_500 app-route-policy test_app_route_policy
```

5. Display the configuration changes:

```
vSmart(config-site-list-site_500)# top
vSmart(config)# show config
```

6. Validate that the configuration contains no errors:

```
vSmart(config)# validate
Validation complete
```

7. Activate the configuration:

```
vSmart(config)# commit
Commit complete.
```

8. Exit from configuration mode:

```
vSmart(config)# exit
vSmart#
```

Full Example Configuration

Putting all the pieces of the configuration together gives this configuration:

```
vSmart# show running-config policy
policy
 sla-class test_sla_class
  latency 50
 !
 app-route-policy test_app_route_policy
  vpn-list vpn_1_list
   sequence 1
    match
     protocol 6
    !
    action sla-class test_sla_class strict
```

```
      !
      sequence 2
       match
        protocol 17
       !
       action sla-class test_sla_class
      !
      sequence 3
       match
        protocol 1
       !
       action sla-class test_sla_class strict
      !
     !
    !
    lists
     vpn-list vpn_1_list
      vpn 1
     !
     site-list site_500
      site-id 500
     !
     site-list site_600
      site-id 600
     !
    !
!
apply-policy
 site-list site_500
  app-route-policy test_app_route_policy
 !
!
```

Additional Information

Application-Aware Routing
Configuring Application-Aware Routing

Service Chaining

Service chaining allows data traffic to be rerouted through one or more services, such as firewall, load balancer, and intrusion detection and prevention (IDP) devices.

Services in the Network

Services such as firewall, load balancer, and intrusion detection and prevention (IDP) are often run within a virtualized environment, and they may physically be centralized in one location or in several locations for redundancy. Services may be internal, cloud based, or external subscriptions. Networks must be able to reroute traffic from any location in the network through such services.

Customers want the ability to internally spawn or externally subscribe to new services on demand—for capacity, redundancy, or simply to select best-of-breed technologies. For example, if a firewall site exceeds its capacity, a customer can spawn new firewall service at a new location. Supporting this new firewall would require the configuration of policy-based, weighted load distribution to multiple firewalls.

Following are some of the reasons to reroute a traffic flow through a service or chain of services:

- Traffic flow from a less secure region of a network must pass through a service, such as a firewall, or through a chain of services to ensure that it has not been tampered with.

- For a network that consists of multiple VPNs, each representing a function or an organization, traffic between VPNs must traverse through a service, such as a firewall, or through a chain of services. For example, in a campus, interdepartmental traffic might go through a firewall, while intradepartmental traffic might be routed directly.
- Certain traffic flows must traverse a service, such as a load balancer.

Today, the only way to reroute traffic flow is by provisioning every routing node—from the source to the service node to the systems beyond the service node—with a policy route. This is done either by having an operator manually configure each node or by using a provisioning tool that performs the configuration for each node on behalf of the operator. Either way, the process is operationally complex to provision, maintain, and troubleshoot.

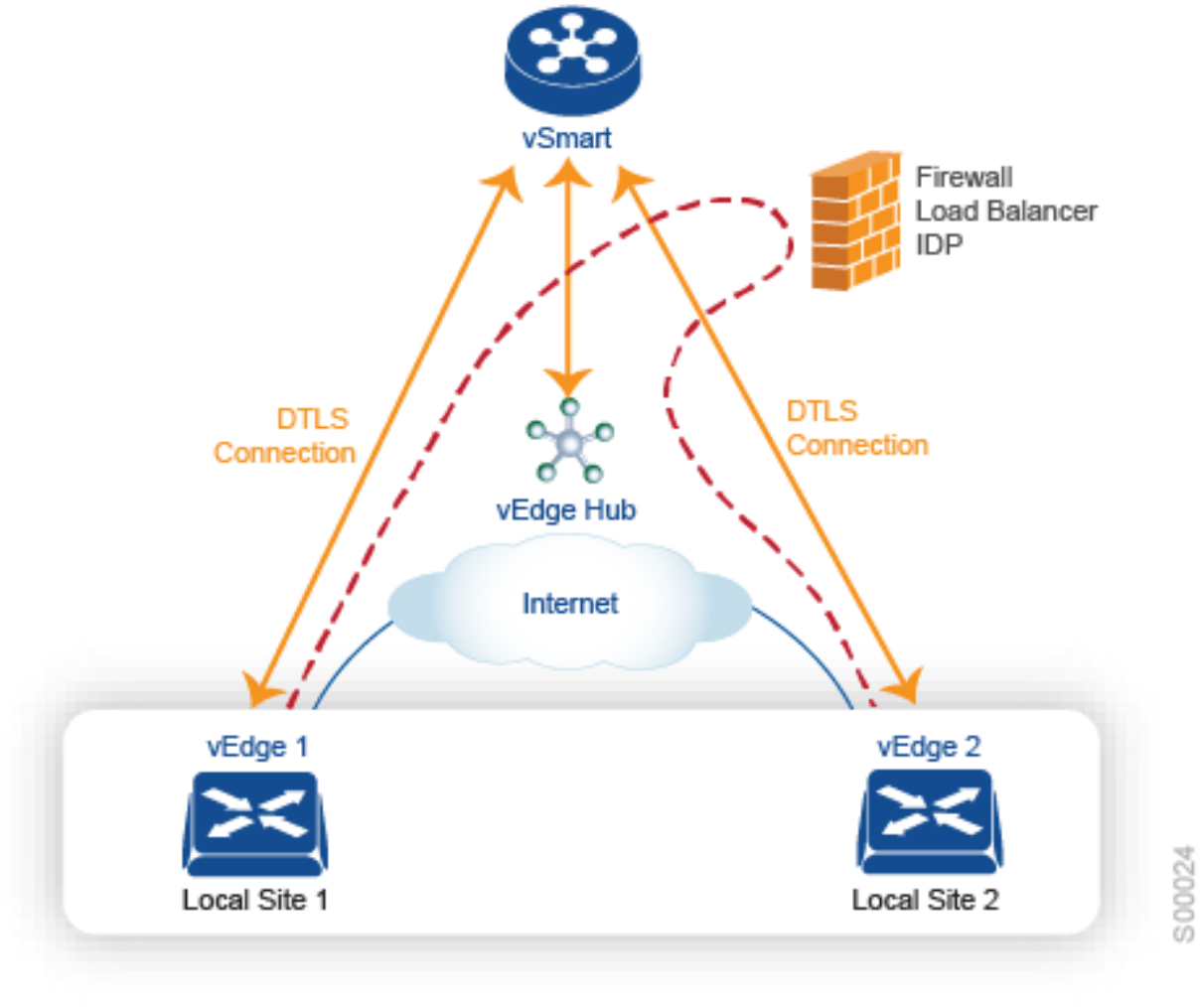
Provisioning Services in the Viptela Overlay Network

In the Viptela solution, the network operator can enable and orchestrate all service chaining from a central controller, that is, from the vSmart controller. No configuration or provisioning is required at any of the vEdge routers.

The general flow of service chaining in a Viptela network is as follows:

- vEdge routers advertise the services available in their branch or campus—such as firewall, IDS, and IDP—to the vSmart controllers in their domain. Multiple vEdge routers can advertise the same services.
- vEdge routers also advertise their OMP routes and TLOCs to the vSmart controllers.
- For traffic that require services, policy on the vSmart controller changes the next hop for the OMP routes to the service landing point. In this way, the traffic is first processed by the service before being routed to its final destination.

The following figure illustrates how service chaining works in the Viptela solution. The network shown has a centralized vEdge hub router that is connected to two branches, each with a vEdge router. The standard network design implements a control policy such that all traffic from branch site 1 to branch site 2 travels through the vEdge hub router. Sitting behind the hub router is a firewall device. So now, assume we want all traffic from site 1 to site 2 to first be processed by the firewall. Traffic from the vEdge router at site 1 still flows to the vEdge hub router, but instead of sending it directly to site 2, the hub router redirects the traffic to the firewall device. When the firewall completes its processing, it returns all cleared traffic to the hub, which then passes it along to the vEdge router at site 2.



Service Route SAFI

The hub and local branch vEdge routers advertise the services available in their networks to the vSmart controllers in its domain using service routes, which are sent via OMP using the service route Subsequent Address Family Identifier (SAFI) bits of the OMP NLRI. The vSmart controllers maintain the service routes in their RIB, and they do not propagate these routes to the vEdges.

Each service route SAFI has the following attributes:

- VPN ID (vpn-id)—Identifies the VPN that the service belongs to.
- Service ID (svc-id)—Identifies the service being advertised by the service node. The Viptela software has the following predefined services:
  - FW, for firewall (maps to svc-id 1)
  - IDS, for Intrusion Detection Systems (maps to svc-id 2)
  - IDP, for Identity Providers (maps to svc-id 3)
  - netsvc1, netsvc2, netsvc3, and netsvc4, which are reserved for custom services (they map to svc-id 4, 5, 6, and 7, respectively)

- Label—For traffic that must traverse a service, the vSmart replaces the label in the OMP route with the service label in order to direct the traffic to that service.
- Originator ID (originator-id)—The IP address of the service node that is advertising the service.
- TLOC—The transport location address of the vEdge that is “hosting” the service.
- Path ID (path-id)—An identifier of the OMP path.

Service Chaining Policy

To route trafic through a service, you provision either a control policy or a data policy on the vSmart controller. You use a control policy if the match criteria are based on a destination prefix or any of its attributes. You use a data policy if the match criteria include the source address, source port, DSCP value, or destination port of the packet or traffic flow. You can provision the policy directly via CLI, or it can be pushed from the vManage management system.

The vSmart controller maintains OMP routes, TLOC routes, and service routes in its route table. A given OMP route carries a TLOC and the label associated with it. On a vSmart controller, a policy can be applied that changes the TLOC and its associated label to be that of a service.

Additional Information

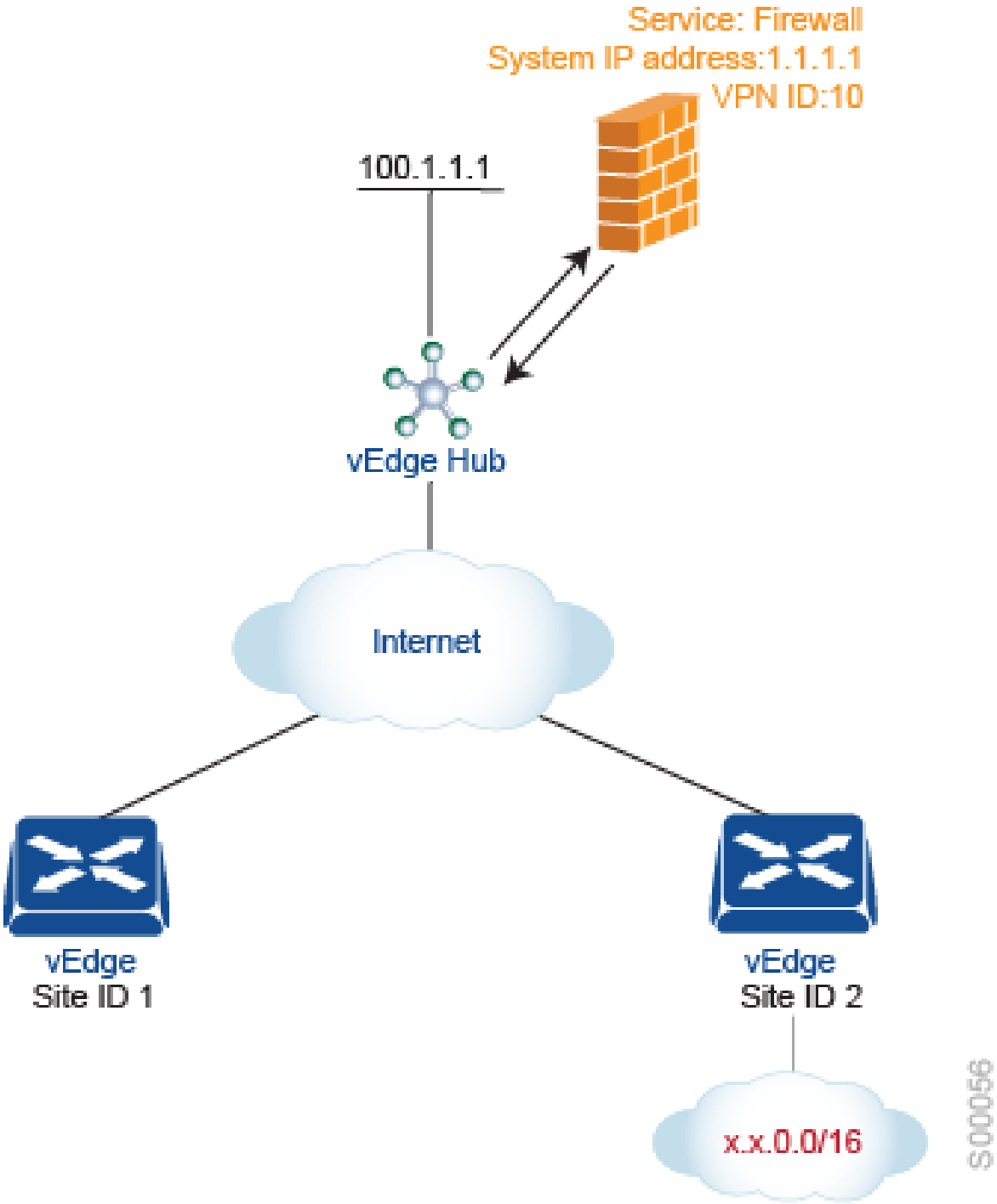
Configuring Centralized Control Policy
Service Chaining Configuration Examples

Service Chaining Configuration Examples

Service chaining control policies direct data traffic to service devices that can be located in various places in the network before the traffic is delivered to its destination. For service chaining to work, you configure a centralized control policy on the vSmart controller, and you configure the service devices themselves on the vEdge router collocated in the same site as the device. To ensure that the services are advertised to the vSmart controller, the IP address of the service device must resolve locally.

This article provides examples of configuring service chaining.

Route Intersite Traffic through a Service



A simple example is to route data traffic traveling from one site to another through a service. In this example, we route all traffic traveling from the vEdge router at Site 1 to the vEdge router at Site 2 through a firewall service that sits behind a vEdge hub (whose system IP address is 100.1.1.1). To keep things simple, all devices are in the same VPN.

For this scenario, you configure the following:

- On the vEdge hub router, you configure the IP address of the firewall device.
- On the vSmart controller, you configure a control policy that redirects traffic destined from Site 1 to Site 2 through the firewall service.
- On the vSmart controller, you apply the control policy to Site 1.

Here is the configuration procedure:

1. On the vEdge hub router, provision the firewall service, specifying the IP address of the firewall device. With this configuration, OMP on the vEdge hub router advertises one service route to the vSmart controller. The service route contains a number of properties that identify the location of the firewall, including the TLOC of the vEdge hub router and a service label of svc-id-1, which identifies the service type as a firewall. (As mentioned above, before advertising the route, the vEdge router ensures that the firewall's IP address can be resolved locally.)

```
vpn 10
  service FW address 1.1.1.1
```

2. On the vSmart controller, configure a control policy that redirects data traffic traveling from Site 1 to Site 2 through the firewall. Then, also on the vSmart controller, apply this policy to Site 1.

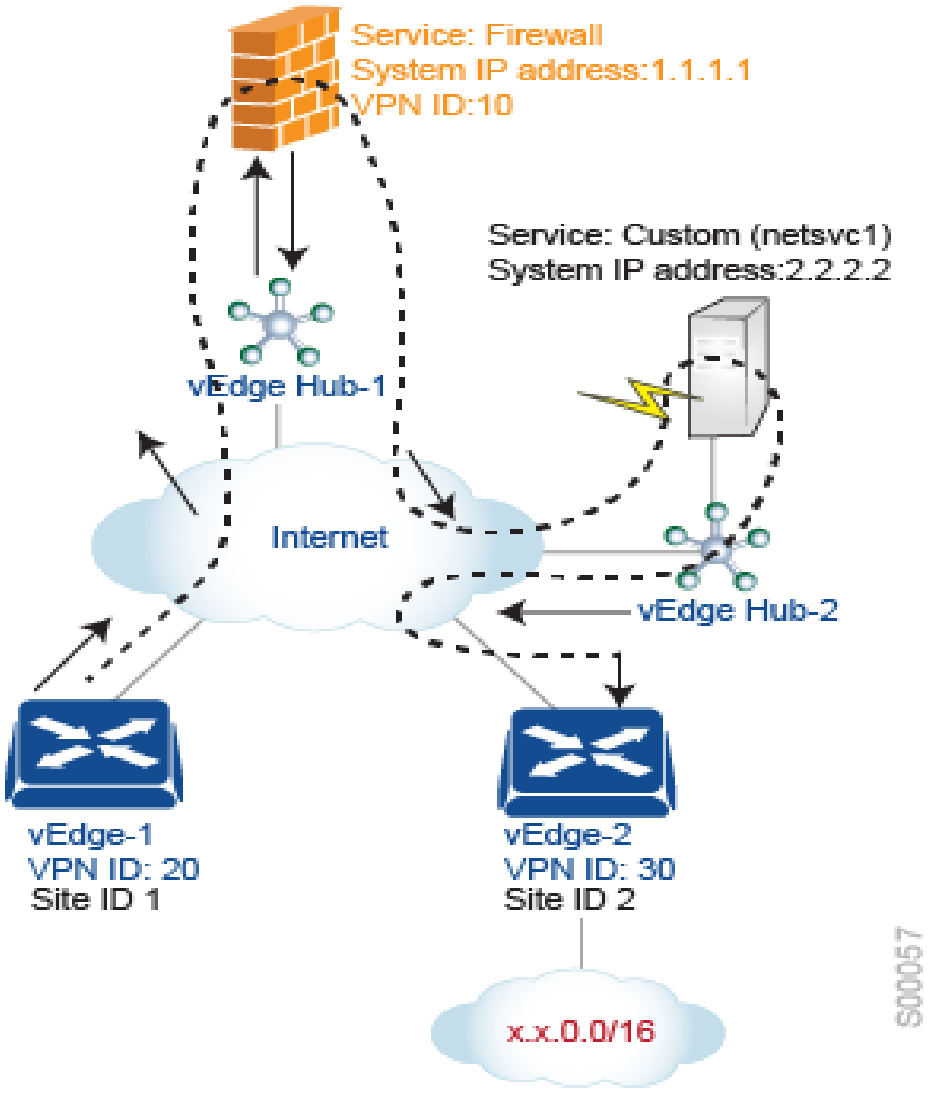
```
policy
  lists
    site-list firewall-sites
      site-id 1
  control-policy firewall-service
    sequence 10
      match route
        site-id 2
      action accept
        set service FW vpn 10
    default-action accept
apply-policy
  site-list firewall-sites control-policy firewall-service out
```

This policy configuration does the following:

- Create a site list called **firewall-sites** that is referenced in the **apply-policy** command and that enumerates all the sites that this policy applies to. If you later want to scale this policy so that all traffic destined to Site 2 from other sites should also first pass through the firewall, all you need to do is add the additional site IDs to the **firewall-sites** site list. You do not need to change anything in the **control-policy firewall-service** portion of the configuration.
- Define a control policy named **firewall-service** . This policy has one sequence element and the following conditions:
  - Match routes destined for Site 2.
  - If a match occurs, accept the route and redirect it to the firewall service provided by the vEdge Hub router, which is located in VPN 10.
  - Accept all nonmatching traffic. That is, accept all traffic not destined for Site 2.
- Apply the policy to the sites listed in **firewall-list** , that is, to Site 1. The vSmart controller applies the policy in the outbound direction, that is, on routes that it redistributes to Site 1. In these routes:
  - The TLOC is changed from Site 2's TLOC to the vEdge hub router's TLOC. This is the TLOC that the vSmart controller learned from the service route received from the vEdge hub router. It is because of the change of TLOC that traffic destined for Site 2 is directed to the vEdge hub router
  - The label is changed to svc-id-1, which identifies the firewall service. This label causes the vEdge hub router to direct the traffic to the firewall device.

When the vEdge hub router receives the traffic, it forwards it to the address 1.1.1.1, which is the system IP address of the firewall. After the firewall has finished processing the traffic, the firewall returns the traffic to the vEdge hub router, and this router then forwards it to its final destination, which is Site 2.

Route Inter-VPN Traffic through a Service Chain with One Service per Node



A service chain allows traffic to pass through two or more services before reaching its destination. The example here routes traffic from one VPN to another through services located in a third VPN. The services are located behind different vEdge hub routers. Specifically, we want all traffic from vEdge-1 in VPN 20 and that is destined for prefix x.x.0.0/16 in VPN 30 on vEdge-2 to go first through the firewall behind vEdge Hub-1 and then through the custom service netsvc1 behind vEdge Hub-2 before being sent to its final destination.

For this policy to work:

- VPN 10, VPN 20, and VPN 30 must be connected by an extranet, such as the Internet
- VPN 10 must import routes from VPN 20 and VPN 30. Routes can be selectively imported if necessary.
- VPN 20 must import routes from VPN 30. Routes can be selectively imported if necessary.
- VPN 30 must import routes from VPN 20. Routes can be selectively imported if necessary.

For this scenario, you configure four things:

- You configure the IP address of the firewall device on the vEdge Hub-1 router.
- You configure the IP address of the custom service device on the vEdge Hub-2 router.
- On the vSmart controller, you configure a control policy that redirects traffic destined from Site 1 to Site 2 through the firewall device.
- On the vSmart controller, you configure a second control policy that redirects traffic to the custom service device.

Here is the configuration procedure:

1. Configure the firewall service on vEdge Hub-1. With this configuration, OMP on the vEdge Hub-1 router advertises a service route to the vSmart controller. The service route contains a number of properties that identify the location of the firewall, including the TLOC of the vEdge hub router and a service label of svc-id-1, which identifies the service type as a firewall.

```
vpn 10
  service fw address 1.1.1.1
```

2. Configure the custom service netsvc1 on vEdge Hub-2. With this configuration, OMP on the vEdge Hub-2 router advertises a service route to the vSmart controller. The service route contains the TLOC of the vEdge Hub-2 and a service label of svc-id-4, which identifies the custom service.

```
vpn 10
  service netsvc1 address 2.2.2.2
```

3. Create a control policy on the vSmart controller for first service in the chain—the firewall—and apply it to Site 1, which is the location of the vEdge-1 router:

```
policy
  lists
    site-list firewall-custom-service-sites
      site-id 1
  control-policy firewall-service
    sequence 10
      match route
        vpn 30
        site-id 2
      action accept
        set service FW
    default-action accept
apply-policy
  site-list firewall-custom-service-sites control-policy firewall-service out
```

This policy configuration does the following:

- Create a site list called **firewall-custom-service-sites** that is referenced in the **apply-policy** command and that enumerates all the sites that this policy applies to.
- Define a control policy named **firewall-service** that has one sequence element and the following conditions:
  - Match routes destined for both VPN 30 and Site 2.
  - If a match occurs, accept the route and redirect it to a firewall service.
  - If a match does not occur, accept the traffic.
- Apply the policy to the sites in the **firewall-custom-service-sites** site list, that is, to Site 1. The vSmart controller applies this policy in the outbound direction, that is, on routes that it redistributes to Site 1. In these routes:
  - The TLOC is changed from Site 2's TLOC to the vEdge Hub-1 router's TLOC. This is the TLOC that the vSmart controller learned from the service route received from the vEdge hub. It is because of the change of TLOC that traffic destined for Site 2 is directed to the vEdge Hub-1 router.
  - The label is changed to svc-id-1, which identifies the firewall service. This label causes the vEdge Hub-2 router to direct the traffic to the firewall device.

When the vEdge Hub-1 router receives the traffic, it forwards it to the address 1.1.1.1, which is the system IP address of the firewall. After the firewall completes processing the traffic, it returns the traffic to the vEdge Hub-1 router, which, because of the policy defined in the next step, forwards it to the vEdge Hub-2 router.

4. Create a control policy on the vSmart controller for the second service in the chain, which is the custom service, and apply it to Site 3, which is the location of the vEdge Hub-2 router:

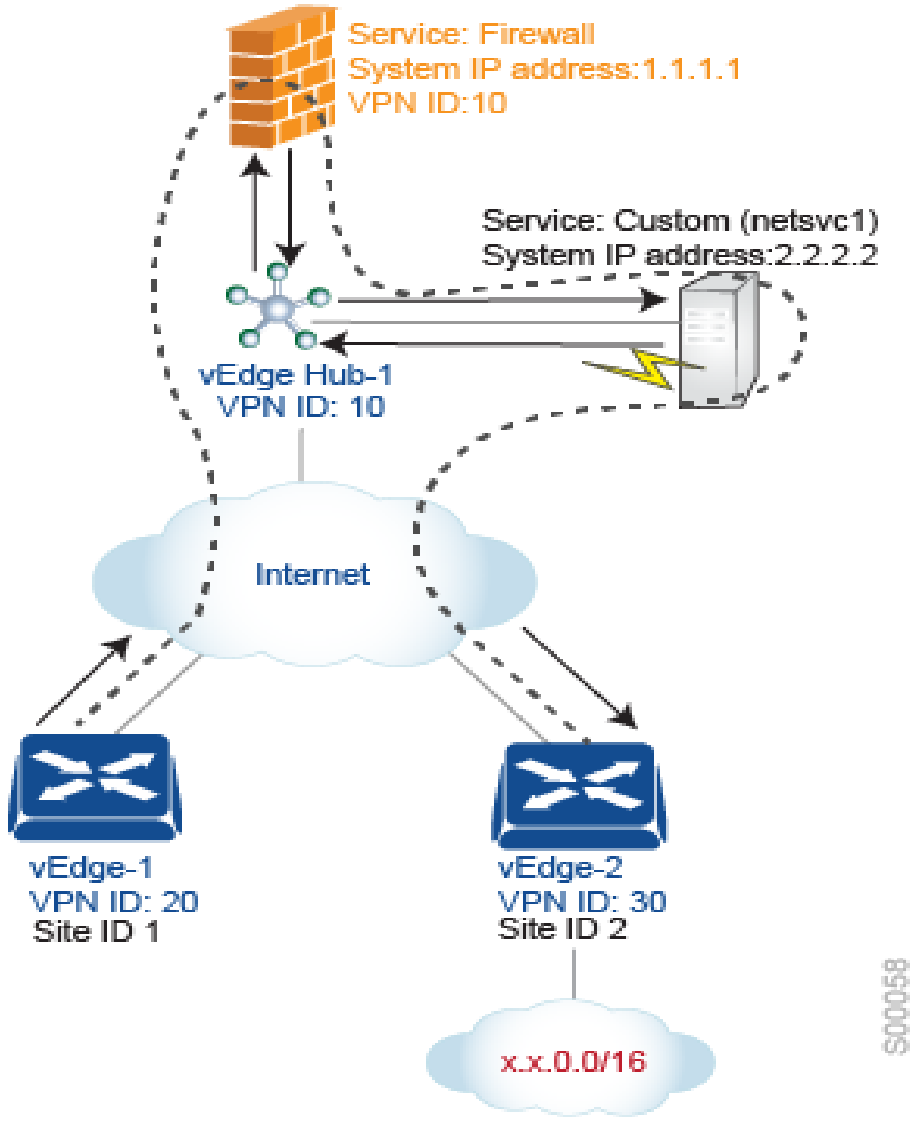
```
policy
  site-list custom-service
    site-id 3
  control-policy netsvc1-service
    sequence 10
      match route
        vpn 10
        site-id 2
      action accept
        set service netsvc1
    default-action accept
apply-policy
  site-list custom-service control-policy netsvc1-service out
```

This policy configuration does the following:

- Create a site list called **custom-service** that is referenced in the **apply-policy** command and that enumerates all the sites that this policy applies to.
- Define a control policy named **netsvc1-service** that has one sequence element and the following conditions:
  - Match routes destined for both VPN 30 and Site 2.
  - If a match occurs, accept the route and redirect it to the custom service.
  - If a match does not occur, accept the traffic.
- Apply the policy to the sites in the **custom-service** list, that is, to Site 3. The vSmart controller applies this policy in the outbound direction, that is, on routes that it redistributes to Site 3. In these routes:
  - The TLOC is changed from Site 2's TLOC to the vEdge Hub-2 router's TLOC. This is the TLOC that the vSmart controller learned from the service route received from the vEdge Hub-2 router. It is because of the change of TLOC that traffic destined for Site 2 is directed to the vEdge Hub-2 router.
  - The label is changed to svc-id-4, which identifies the custom service. This label causes the vEdge Hub-2 to direct the traffic to the device that is hosting the custom service

When the vEdge Hub-2 routers receives the traffic, it forwards it to the address 2.2.2.2, which is the system IP address of the device hosting the custom service. After the traffic has been processed, it is returned to the vEdge Hub-2 router, which then forwards it to its final destination, Site 2.

Route Inter-VPN Traffic through a Service Chain with Multiple Services per Node



If a service chain has more than one service that is connected to the same node, that is, both services are behind the same vEdge router, you use a combination of control policy and data policy to create the desired service chain. The example here is similar to the one in the previous section, but instead has a firewall and a custom service (netsvc-1) behind a single vEdge hub router. Here, we want all data traffic from vEdge-1 in VPN 20 destined for prefix x.x.0.0/16 on vEdge-2 in VPN 30 to first go through the firewall at vEdge Hub-1, then through the custom service netsvc1, also at vEdge Hub1, and then to its final destination.

For this policy to work:

- VPN 10, VPN 20, and VPN 30 must be connected by an extranet, such as the Internet.
- VPN 10 must import routes from VPN 20 and VPN 30. Routes can be selectively imported if necessary.
- VPN 20 must import routes from VPN 30. Routes can be selectively imported if necessary.
- VPN 30 must import routes from VPN 20. Routes can be selectively imported if necessary.

For this scenario, you configure the following:

- On the vEdge hub router, you configure the firewall and custom services.
- On the vSmart controller, you configure a control policy that redirects data traffic from Site 1 that is destined to Site 2 through the firewall.
- On the vSmart controller, you configure a data policy that redirects data traffic to the custom service.

Here is the configuration procedure:

1. On the vEdge hub router, configure the firewall and custom services:

```
vpn 10
  service FW address 1.1.1.1
  service netsvc1 address 2.2.2.2
```

With this configuration, OMP on the vEdge hub router advertises two service routes to the vSmart Control, one for the firewall and the second for the custom service netsvc1. Both service routes contain the TLOC of the vEdge Hub-1 router and a service label that identifies the type of service. For the firewall service, the label is svc-id-1, and for the custom service, the label is svc-id-4.

2. On the vSmart controller, configure a control policy controller to reroute traffic destined for VPN 30 (at Site 2) to firewall service that is connected to vEdge Hub-1 (at Site 3), and apply this policy to Site 1:

```
policy
  lists
    site-list vEdge-1
      site-id 1
  control-policy firewall-service
    sequence 10
      match route
        vpn 30
      action accept
        set service FW
apply-policy
  site-list vEdge-1 control-policy firewall-service out
```

3. On the vSmart controller, configure a data policy that redirects, or chains, the data traffic received from the firewall device to the custom service netsvc1. Then apply this policy to vEdge Hub-1. This data policy routes packets headed for destinations in the network x.x.0.0/16 to the IP address 2.2.2.2, which is the system IP address of the device hosting the custom service.

```
policy
  lists
    site-list vEdge-2
      site-id 2
    site-list vEdge-Hub-1
      site-id 3
    prefix-list svc-chain
      ip-prefix x.x.0.0/16
    vpn-list vpn-10
      vpn 10
  data-policy netsvc1-policy
    vpn-list vpn-10
      sequence 1
        match
          ip-destination x.x.0.0/16
        action accept
          set next-hop 2.2.2.2
apply-policy
  site-list vEdge-Hub-1 data-policy netsvc1-policy from-service
```

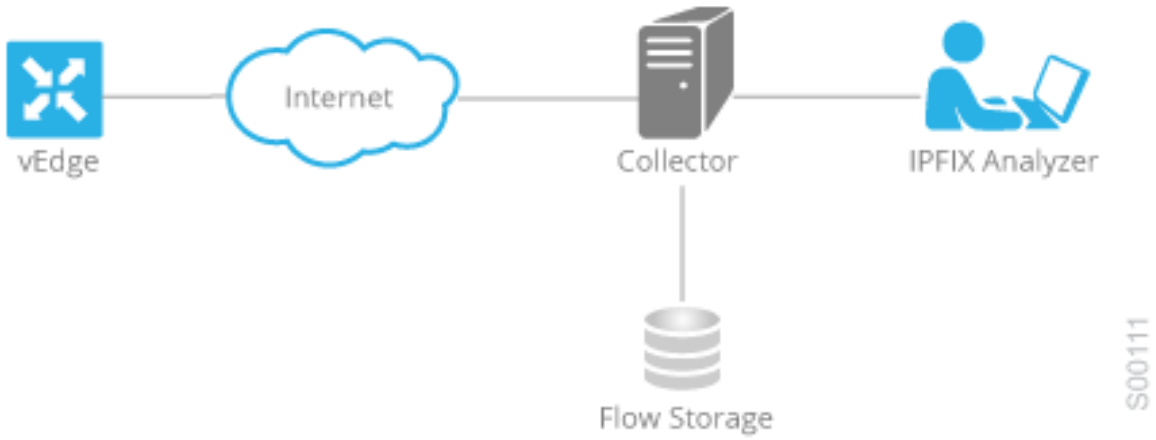
Additional Information

Service Chaining

Traffic Flow Monitoring with Cflowd

Cflowd monitors traffic flowing through vEdge routers in the overlay network and exports flow information to a collector, where it can be processed by an IPFIX analyzer. For a traffic flow, cflowd periodically sends template reports to flow collector. These reports contain information about the flow and data extracted from the IP headers of the packets in the flow.

The Viptela cflowd software implements cflowd version 10, as specified in RFC 7011 and RFC 7012 . Cflowd version 10 is also called the IP Flow Information Export (IPFIX) protocol.



Cflowd performs 1:1 sampling. Information about all flows is aggregated in the cflowd records; flows are not sampled. vEdge routers do not cache any of the records that are exported to a collector.

Components of Cflowd

In the Viptela overlay network, you configure cflowd using centralized data policy. As part of the policy, you specify the location of the collector. By default, flow information is sent to the collector every 60 seconds. You can modify this and other timers related to how often cflowd templates are refreshed and how often a traffic flow times out.

You can configure a maximum of four cflowd policies. The Viptela software can export template records to a maximum of four cflowd collectors. When you configure a new data policy that changes which flows are sampled, the software allows the old flows to expire gracefully rather than deleting them all at once.

The vEdge router exports template records and data records to a collector. The template record is used by the collector to parse the data record information that is exported to it. Option templates are not supported. The source IP address for the packet containing the IPFIX records is randomly selected from any of the interfaces in the VPN. The flow records are exported via TCP or UDP connections. Anonymization of records and TLS encryption are not performed, because it is assumed that the collector and the IPFIX analyzer are both located within the data center, traffic traveling within the data center is assumed to be safe.

Cflowd can track GRE, ICMP, IPsec, SCTP, TCP, and UDP flows.

IPFIX Information Elements Exported to the Collector

The Viptela cflowd software exports the following 22 IPFIX information elements to the cflowd collector. These information elements are a subset of those defined in RFC 7012 and maintained by IANA . The elements are exported in the order listed. You cannot modify the information elements that are exported, nor can you change the order in which they appear.

Information Element	Element ID	Description	Data Type	Data Type Semantics	Units or Range
VPN Identifier	Enterprise specific	Viptela VPN identifier. Viptela uses the enterprise ID for VIP_IANA_ENUM or 41916, and the VPN element ID is 4321.	unsigned32 (8 bytes)	identifier	0 through 65535
sourceIPv4Address	8	IPv4 source address in the IP packet header.	ipv4Address (4 bytes)	default	—
destinationIPv4Address	12	IPv4 destination address in the IP packet header.	IPv4Address (4 bytes)	default	—
ipDiffServCodePoint	195	Value of a Differentiated Services Code Point (DSCP) encoded in the Differentiated Services field. This field spans the most significant 6 bits of the IPv4 TOS field.	unsigned8 (1 byte)	identifier	0 through 63
destinationTransportPort	11	Destination port identifier in the transport header. For the transport protocols UDP, TCP, and SCTP, this is the destination port number given in the respective header.	unsigned16 (2 bytes)	identifier	—
sourceTransportPort	7	Source port identifier in the transport header. For the transport protocols UDP, TCP, and SCTP, this is the destination port number given in the respective header. For GRE and IPsec flows, the value of this field is 0.	unsigned16 (2 bytes)	identifier	—
protocolIdentifier	4	Value of the protocol number in the Protocol field of the IP packet header. The protocol number identifies the IP packet payload type. Protocol numbers are defined in the IANA Protocol Numbers registry.	unsigned8 (1 byte)	identifier	—
flowStartSeconds	150	Absolute timestamp of the first packet of this flow.	dateTime-Seconds (4 bytes)	—	—
flowEndSeconds	151	Absolute timestamp of the last packet of this flow.	dateTime-Seconds (4 bytes)	—	—
octetTotalCount	85	Total number of octets in incoming packets for this flow at the observation point since initialization or re-initialization of the metering process for the observation point. The count includes the IP headers and IP payload.	unsigned64 (8 bytes)	totalCounter	Octets
octetDeltaCount	1	Number of octets since the previous report in incoming packets for this flow at the observation point. This number includes IP headers and IP payload.	unsigned 64 (8 bytes)	deltaCounter	Octets
packetTotalCount	86	Total number of incoming packets for this flow at the observation point since initialization or re-initialization of the metering process for the observation point.	unsigned64 (8 bytes)	totalCounter	Packets

packetDeltaCount	2	Number of incoming packets since the previous report for this flow at this observation point.	unsigned64 (8 bytes)	deltaCounter	Packets
tcpControlBits	6	TCP control bits observed for the packets of this flow. This information is encoded as a bit field; each TCP control bit has a bit in this set. The bit is set to 1 if any observed packet of this flow has the corresponding TCP control bit set to 1. Otherwise, the bit is set to 0. For values of this field, see the IANA IPFIX web page .	unsigned16 (2 bytes)	flags	—
maximumIpTotalLength	26	Length of the largest packet observed for this flow. The packet length includes the IP headers and IP payload.	unsigned64 (8 bytes)	—	Octets
minimumIpTotalLength	25	Length of the smallest packet observed for this flow. The packet length includes the IP headers and IP payload.	unsigned64 (8 bytes)	—	Octets
ipNextHopIPv4Address	15	IPv4 address of the next IPv4 hop.	IPv4Address (4 bytes)	default	—
egressInterface	14	Index of the IP interface where packets of this flow are being sent.	unsigned32 (8 bytes)	default	—
ingressInterface	10	Index of the IP interface where packets of this flow are being received.	unsigned32 (8 bytes)	identifier	—
icmpTypeCodeIPv4	32	Type and Code of the IPv4 ICMP message. The combination of both values is reported as (ICMP type * 256) + ICMP code.	unsigned16 (4 bytes)	identifier	—
flowEndReason	136	Reason for the flow termination. For values of this field, see the IANA IPFIX web page	unsigned8 (1 byte)	identifier	—
ipClassOfService	5	Value of type of service (TOS) field in the IPv4 packet header.	unsigned8 (1 byte)	identifier	—
ipPrecedence	196	Value of IP precedence. This value is encoded in the first 3 bits of the IPv4 TOS field.	unsigned8 (1 byte)	flags	0 through 7
paddingOctets	210	Value of this Information Element is always a sequence of 0x00 values.	octetArray	default	—

Additional Information

Cflowd Traffic Flow Monitoring Configuration Example
Configuring Cflowd Traffic Flow Monitoring

Configuring Cflowd Traffic Flow Monitoring

This article provides general procedures for configuring cflowd traffic flow monitoring.

You configure cflowd traffic flow monitoring using the basic components of centralized data policy. You configure cflowd template options, including the location of the cflowd collector (if you are sending the flow to a collector), and you must configure cflowd as an action in the data policy.

General vManage Configuration Procedure for Cflowd Traffic Flow Monitoring

To configure policy for cflowd traffic flow monitoring, use the vManage policy configuration wizard. The wizard consists of four sequential screens that guide you through the process of creating and editing policy components:

1. Create Applications or Groups of Interest—Create lists that group together related items and that you call in the match or action components of a policy.
2. Configure Topology—Create the network structure to which the policy applies.
3. Configure Traffic Rules—Create the match and action conditions of a policy.
4. Apply Policies to Sites and VPNs—Associate policy with sites and VPNs in the overlay network.

In the first three policy configuration wizard screens, you are creating policy components or blocks. In the last screen, you are applying policy blocks to sites and VPNs in the overlay network.

For the cflowd policy to take effect, you must activate the policy.

Start the Policy Configuration Wizard

To start the policy configuration wizard:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy.

The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed.

Create Applications or Groups of Interest

To create lists of applications or groups to use in cflowd policy:

1. Start the policy configuration wizard as explained above.
2. Create new lists , as described in the following table:

List Type	Procedure
Prefix	1. In the left bar, click Prefix. 2. Click New Prefix List. 3. Enter a name for the list. 4. In the Add Prefix field, enter one or more data prefixes separated by commas. 5. Click Add.
Site	6. In the left bar, click Site. 7. Click New Site List. 8. Enter a name for the list. 9. In the Add Site field, enter one or more site IDs separated by commas. 10. Click Add.

VPN	11. In the left bar, click VPN. 12. Click New VPN List. 13. Enter a name for the list. 14. In the Add VPN field, enter one or more VPN IDs separated by commas. 15. Click Add.

4. Click Next to move to Configure Topology in the wizard. When you first open this screen, the Topology tab is selected by default.

Configure the Network Topology

To configure the network topology or a VPN membership to use in centralized policy:

1. If you are already in the policy configuration wizard, skip to Step 4. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Create a network topology, as described in the following table:

Policy Type	Description	Procedure
Hub and Spoke	Policy for a topology with one or more central hub sites and with spokes connected to a hub	1. In the Add Topology drop-down, select Hub and Spoke. 2. Enter a name for the hub-and-spoke policy. 3. Enter a description for the policy. 4. In the VPN List field, select the VPN list for the policy. 5. In the left pane, click Add Hub and Spoke. A hub-and-spoke policy component containing the text string My Hub-and-Spoke is added in the left pane. 6. Double-click the My Hub-and-Spoke text string, and enter a name for the policy component. 7. In the right pane, add hub sites to the network topology: a. Click Add Hub Sites. b. In the Site List Field, select a site list for the policy component. c. Click Add. d. Repeat Steps 7a, 7b, and 7c to add more hub sites to the policy component. 8. In the right pane, add spoke sites to the network topology: a. Click Add Spoke Sites. b. In the Site List Field, select a site list for the policy component. c. Click Add. d. Repeat Steps 8a, 8b, and 8c to add more spoke sites to the policy component. 9. Repeat Steps 5 through 8 to add more components to the hub-and-spoke policy. 10. Click Save Hub and Spoke Policy.
Mesh	Partial-mesh or full-mesh region	11. In the Add Topology drop-down, select Mesh. 12. Enter a name for the mesh region policy component. 13. Enter a description for the mesh region policy component. 14. In the VPN List field, select the VPN list for the policy. 15. Click New Mesh Region. 16. In the Mesh Region Name field, enter a name for the individual mesh region. 17. In the Site List field, select one or more sites to include in the mesh region. 18. Repeat Steps 5 through 7 to add more mesh regions to the policy. 19. Click Save Mesh Region.

5. To use an existing topology:
    a. In the Add Topology drop-down, click Import Existing Topology. The Import Existing Topology popup displays.
    b. Select the type of topology.
    c. In the Policy drop-down, select the name of the topology.
    d. Click Import.
6. Click Next to move to Configure Traffic Rules in the wizard. When you first open this screen, the Application-Aware Routing tab is selected by default.

Configure Traffic Rules

To create the match and action rules to apply to traffic affected by the policy:

1. If you are already in the policy configuration wizard, skip this procedure. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.

To configure traffic rules for cflowd policy:

1. In the Application-Aware Routing bar, select the Cflowd tab.
2. Click the Add Policy drop-down.
3. Select Create New. The Add Cflowd Policy popup opens.
4. Configure timer parameters for the cflowd template:
    a. In the Active Flow Timeout field, specify how long to collect a set of flows on which traffic is actively flowing, a value from 30 through 3,600 seconds. The default is 600 seconds (10 minutes).
    b. In the Inactive Flow Timeout field, specify how long to wait to send a set of sampled flows to a collector for a flow on which no traffic is flowing, a value from 1 through 3,600 seconds. The default is 60 seconds (1 minute).
    c. In the Flow Refresh Interval field, specify how often to send the cflowd template record fields to the collector, a value from 60 through 86,400 seconds (1 minute through 1 day). The default is 90 seconds.
    d. In the Sampling Interval field, specify how many packets to wait before creating a new flow, a value from 1 through 65,536 seconds. While you can configure any integer value, the software rounds the value down to the nearest power of 2.
5. Click Add New Collector, and configure the location of the cflowd collector. You can configure up to four collectors.
    a. In the VPN ID field, enter the number of the VPN in which the collector is located.
    b. In the IP Address field, enter the IP address of the collector.
    c. In the Port Number field, enter the collector port number. The default port is 4739.
    d. In the Transport Protocol drop-down, select the transport type to use to reach the collector, either TCP or UDP.

e. In the Source Interface field, enter the name of the interface to use to send flows to the collector. It can be either a Gigabit Ethernet, a 10-Gigabit Ethernet interface (**ge**), or a loopback interface (**loopback** *number*).

6. Click Save Cflowd Policy.

Click Next to move to Apply Policies to Sites and VPNs in the wizard.

Apply Policies to Sites and VPNs

In the last screen of the policy configuration wizard, you associate the policy blocks that you created on the previous three screens with VPNs and with sites in the overlay network.

To apply a policy block to sites and VPNs in the overlay network:

1. If you are already in the policy configuration wizard, skip to Step 6. Otherwise, in vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Click Add Policy. The policy configuration wizard opens, and the Create Applications or Groups of Interest screen is displayed
3. Click Next. The Network Topology screen opens, and in the Topology bar, the Topology tab is selected by default.
4. Click Next. The Configure Traffic Rules screen opens, and in the Application-Aware Routing bar, the Application-Aware Routing tab is selected by default.
5. Click Next. The Apply Policies to Sites and VPNs screen opens.
6. In the Policy Name field, enter a name for the policy. This field is mandatory and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.
7. In the Policy Description field, enter a description of the policy. It can contain up to 2048 characters. This field is mandatory, and it can contain any characters and spaces.
8. From the Topology bar, select the type of policy block. The table then lists policies that you have created for that type of policy block.
9. Click Add New Site List. Select one or more site lists, Click Add.
10. Click Preview to view the configured policy. The policy is displayed in CLI format.
11. Click Save Policy. The Configuration ► Policies screen opens, and the policies table includes the newly created policy.

Activate a Centralized Policy

Activating a cflowd policy sends that policy to all connected vSmart controllers. To activate a cflowd policy:

1. In vManage NMS, select the Configure ► Policies screen. When you first open this screen, the Centralized Policy tab is selected by default.
2. Select a policy.
3. Click the More Actions icon to the right of the row, and click Activate. The Activate Policy popup opens. It lists the IP addresses of the reachable vSmart controllers to which the policy is to be applied.
4. Click Activate.

General Cflowd Routing Policy CLI Configuration Procedure

Following are the high-level steps for configuring a cflowd centralized data policy to perform traffic monitoring and to export traffic flows to a collector:

1. Create a list of overlay network sites to which the cflowd centralized data policy is to be applied (in the **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists- *list-name*)#& **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–). Create additional site lists, as needed.

2. Create a list of VPN for which the cflowd centralized data policy is to be configured (in the **policy data-policy** command):
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists- *list-name*)# **vpn** *vpn-id*

3. Create lists of IP prefixes, as needed:
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **prefix-list** *list-name*
   vSmart(config-lists- *list-name*)# **ip-prefix** *prefix* / *length*

4. Configure a cflowd template, and optionally, configure template parameters, including the location of the cflowd collector, the flow export timers, and the flow sampling interval:
   vSmart(config)# **policy cflowd-template** *template-name*
   vSmart(config-cflowd-template- *template-name*)# **collector vpn** *vpn-id* **address** *ip-address* **port** *port-number* **transport-type** (**transport_tcp** | **transport_udp**) **source-interface** *interface-name*
   vSmart(config-cflowd-template- *template-name*)# **flow-active-timeout** *seconds*
   vSmart(config-cflowd-template- *template-name*)# **flow-inactive-timeout** *seconds*
   vSmart(config-cflowd-template- *template-name*)# **flow-sampling-interval** *number*
   vSmart(config-cflowd-template- *template-name*)# **template-refresh** *seconds*
   You must configure a cflowd template, but it need not contain any parameters. With no parameters, the data flow cache on vEdge nodes is managed using default settings, and no flow export occurs.
   You can configure one cflowd template per vEdge router, and it can export to a maximum of four collectors. By default, an actively flowing data set is exported to the collector every 600 seconds (10 minutes), a data set for a flow on which no traffic is flowing is sent every 60 seconds (1 minute), and the cflowd template record fields (the three timer values) are sent to the collector every 90 seconds. Also by default, a new flow is created immediately after an existing flow has ended.
   If you modify the configuration of the template record fields, the changes take effect only on flows that are created after the configuration change has been propagated to the vEdge router. Because an existing flow continues indefinitely, to have configuration changes take effect, clear the flow with the **clear app cflowd flows** command.

5. If you configure a logging action, configure how often to log packets to the syslog files:
   vEdge(config)# **policy log-frequency** *number*

6. Create a data policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy data-policy** *policy-name*
   vSmart(config-data-policy- *policy-name*)# **vpn-list** *list-name*

7. Create a sequence to contain a single match–action pair:
   vSmart(config-vpn-list- *list-name*)# **sequence** *number*
   vSmart(config-sequence- *number*)#
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. If no match occurs, the default action is taken.

8. Define match parameters for the data packets:
   vSmart(config-sequence- *number*)# **match** *parameters*

9. In the action, enable cflowd:
   vSmart(config-sequence- *number*)# **action cflowd**

10. In the action, count or log data packets:
    vSmart(config-sequence- *number*)# **action count** *counter-name*
    vSmart(config-sequence- *number*)# **action log**

11. Create additional numbered sequences of match–action pairs within the data policy, as needed.

12. If a route does not match any of the conditions in one of the sequences, it is rejected by default. If you want nonmatching prefixes to be accepted, configure the default action for the policy:
    vSmart(config- *policy-name*)# **default-action accept**

13. Apply the policy and the cflowd template to one or more sites in the overlay network:
    vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name*
    vSmart(config)# **apply-policy site-list** *list-name* **cflowd-template** *template-name*

Structural Components of Policy Configuration for Cflowd

Here are the structural components required to configure cflowd on a vSmart controller. Each component is explained in more detail in the sections below.

```
policy
  lists
    prefix-list list-name
      ip-prefix prefix
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn-id vpn-id
  log-frequency number
  cflowd-template template-name
    collector vpn vpn-id address ip-address port port-number transport transport-type source-interface
interface-name
    flow-active-timeout seconds
    flow-inactive-timeout seconds
    flow-sampling-interval number
    template-refresh seconds
  data-policy policy-name
    vpn-list list-name
      sequence number
        match
          match-parameters
        action
          cflowd
          count counter-name
          drop
          log
       default-action
         (accept | drop)
apply-policy site-list list-name
  data-policy policy-name
  cflowd-template template-name
```

Lists

Centralized data policy uses the following types of lists to group related items. You configure lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command

Data prefixes	List of one or more IP prefixes. To configure multiple prefixes in a single list, include multiple **ip-prefix** options, specifying one prefix in each option.	**data-prefix-list** *list-name* **ip-prefix** *prefix / length*
Sites	List of one or more site identifiers in the overlay network. To configure multiple sites in a single list, include multiple **site-id** options, specifying one site number in each option. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*
VPNs	List of one or more VPNs in the overlay network. To configure multiple VPNs in a single list, include multiple **vpn** options, specifying one VPN number in each option. You can specify a single VPN identifier (such as **vpn-id 1**) or a range of VPN identifiers (such as **vpn-id 1-10**).	**vpn-list** *list-name* **vpn** *vpn-id*

Logging Frequency

If you configure a logging action, by default, the vEdge router logs all data packet headers to a syslog file. To log only a sample of the data packet headers:

```
vEdge(config)# policy log-frequency number
```

number specifies how often to to log packet headers. For example, if you configure **log-frequency 20** , every sixteenth packet is logged. While you can configure any integer value for the frequency, the software rounds the value down to the nearest power of 2.

Cflowd Templates

For each cflowd data policy, you must create a template that defines the location of the flow collector:

```
vSmart(config)# policy cflowd-template template-name
```

The template can specify cflowd parameters or it can be empty. With no parameters, the data flow cache on vEdge nodes is managed using default settings, and no flow export occurs.

In the cflowd template, you can define the location of the flow collection:

```
vSmart(config-cflowd-template-template-name)# collector vpn vpn-id address ip-address port port-number
transport transport-type source-interface interface-name
```

You can configure one cflowd template per vEdge router, and it can export to a maximum of four collectors.

You can configure flow export timers:

```
vSmart(config)# policy cflowd-template template-name
vSmart(config-cflowd-template-template-name)# flow-active-timeout seconds
vSmart(config-cflowd-template-template-name)# flow-inactive-timeout seconds
vSmart(config-cflowd-template-template-name)# flow-sampling-interval number
vSmart(config-cflowd-template-template-name)# template-refresh seconds
```

By default, an actively flowing data set is exported to the collector every 600 seconds (10 minutes), a data set for a flow on which no traffic is flowing is sent every 60 seconds (1 minute), and the cflowd template record fields are sent to the collector every 90 seconds. For flow sampling, by default, a new flow is started immediately after an existing flow ends.

For a single vEdge router, you can configure a maximum of four collectors.

Data Policy Instance

For each centralized data policy, you create a named container for that policy with a **policy data-policy** *policy-name* command. For a single vEdge router, you can configure a maximum of four cflowd policies.

VPN Lists

Each centralized data policy instance applies to the VPNs contained in a VPN list. Within the policy, you specify the VPN list with the **policy data-policy vpn-list** *list-name* command. The list name must be one that you created with a **policy lists vpn-list** *list-name* command.

Sequences

Within each VPN list, a centralized data policy contains sequences of match–action pairs. The sequences are numbered to set the order in which data traffic is analyzed by the match–action pairs in the policy. You configure sequences with the **policy data-policy vpn-list sequence** command.

Each sequence in a centralized data policy can contain one **match** command and one **action** command.

Match Parameters

Centralized data policy can match IP prefixes and fields in the IP headers. You configure the match parameters under the **policy data-policy vpn-list sequence match** command.

For data policy, you can match these parameters:

Description	Command	Value or Range
Group of destination prefixes	**destination-data-prefix-list** *list-name*	Name of a **data-prefix-list** list.
Individual destination prefix	**destination-ip** *prefix* / *length*	IP prefix and prefix length
Destination port number	**destination-port** *number*	0 through 65535
DSCP value	**dscp** *number*	0 through 63
Internet Protocol number	**protocol** *number*	0 through 255
Group of source prefixes	**source-data-prefix-list** *list-name*	Name of a **data-prefix-list** list
Individual source prefix	**source-ip** *prefix* / *length*	IP prefix and prefix length
Source port number	**source-port** *address*	0 through 255

Action Parameters

When data traffic matches the conditions in the match portion of a centralized data policy, the packet can be accepted or rejected, and you can configure a counter for the accepted or rejected packets. You configure the action parameters under the **policy data-policy vpn-list sequence action** command.

Description	Command	Value or Range
Count the accepted or dropped packets.	**count** *counter-name*	Name of a counter. To display counter information, use the **show policy access-lists counters** command on the vEdge router.
Enable cflowd.	**cflowd**	—

Log the packet headers into the messages and vsyslog system logging (syslog) files. In addition to logging the packet headers, a syslog message is generated the first time a packet header is logged and then every 5 minutes thereafter, as long as the flow is active.	**log**	To display logging information, use the **show app log flow-all** , **show app log flows** , and **show log** commands on the vEdge router.

For a packet that is accepted, configure the parameter **cflowd** to enable packet collection.

Default Action

If a data packet being evaluated does not match any of the match conditions in a control policy, a default action is applied to this route. By default, the route is rejected. To modify this behavior, include the **policy data-policy vpn-list default-action accept** command.

Applying Cflowd Policy

For a centralized data policy to take effect, you must apply it to a list of sites in the overlay network:

```
vSmart(config)# apply-policy site-list list-name data-policy policy-name
```

To activate the cflowd template, associate it with the data policy:

```
vSmart(config)# apply-policy cflowd-template template-name
```

For all **data-policy** policies that you apply with **apply-policy** commands, the site IDs across all the site lists must be unique. That is, the site lists must not contain overlapping site IDs. An example of overlapping site IDs are those in the two site lists **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** . Here, sites 70 through 100 are in both lists. If you were to apply these two site lists to two different **data-policy** policies, the attempt to commit the configuration on the vSmart controller would fail.

The same type of restriction also applies to the following types of policies:

- Application-aware routing policy (**app-route-policy**)
- Centralized control policy (**control-policy**)
- Centralized data policy (**data-policy**)

You can, however, have overlapping site IDs for site lists that you apply for different types of policy. For example, the sites lists for **control-policy** and **data-policy** policies can have overlapping site IDs. So for the two example site lists above, **site-list 1 site-id 1-100** and **site-list 2 site-id 70-130** , you could apply one to a control policy and the other to a data policy.

As soon as you successfully activate the configuration by issuing a **commit** command, the vSmart controller pushes the data policy to the vEdge routers located in the specified sites. To view the policy as configured on the vSmart controller, use the **show running-config** command on the vSmart controller. To view the policy that has been pushed to the vEdge router, use the **show policy from-vsmart** command on the vEdge router.

To display the centralized data policy as configured on the vSmart controller, use the **show running-config** command:

```
vSmart# show running-config policy
vSmart# show running-config apply-policy
```

To display the centralized data policy that has been pushed to the vEdge router, issue the **show omp data-policy** command on the vEdge router:

```
vEdge# show policy from-vsmart
```

Enable Cflowd Visibility on vEdge Routers

You can enable cflowd visibility directly on vEdge routers, without configuring data policy, so that you can perform traffic flow monitoring on traffic coming to the router from all VPNs in the LAN. To do this, configure cflowd visiblity on the router:

```
vEdge(config)# policy flow-visibility
```

To monitor the applications, use the **show app cflowd flows** and **show app cflowd statistics** commands on the vEdge router.

Additional Information

Cflowd Traffic Flow Monitoring Configuration Example
Traffic Flow Monitoring with Cflowd

Cflowd Traffic Flow Monitoring Configuration Example

This article shows a straightforward example of configuring traffic flow monitoring.

Configuration Steps

You enable cflowd traffic monitoring with a centralized data policy, so all configuration is done on a vSmart controller. The following example procedure monitors all TCP traffic, sending it to a single collector:

1. Create a cflowd template to define the location of the collector and to modify cflowd timers:
   vSmart(config)# **policy cflowd-template test-cflowd-template**
   vSmart(config-cflowd-template-test-cflowd-template)# **collector vpn 1 address 172.16.155.15 port 13322 transport transport_udp**
   vSmart(config-cflowd-template-test-cflowd-template)# **flow-inactive-timeout 60**
   vSmart(config-cflowd-template-test-cflowd-template)# **template-refresh 90**

2. Create a list of VPNs whose traffic you want to monitor:
   vSmart(config)# **policy lists vpn-list vpn_1 vpn 1**

3. Create a list of sites to apply the data policy to:
   vSmart(config)# **policy lists site-list cflowd-sites site-id 400,500,600**

4. Configure the data policy itself:
   vSmart(config)# **policy data-policy test-cflowd-policy**
   vSmart(config-data-policy-test-cflowd-policy)# **vpn-list vpn_1**
   vSmart(config-vpn-list-vpn_1)# **sequence 1**
   vSmart(config-sequence-1)# **match protocol 6**
   vSmart(config-match)# **exit**
   vSmart(config-sequence-1)# **action accept cflowd**
   vSmart(config-action)# **exit**
   vSmart(config-sequence-1)# **exit**
   vSmart(config-vpn-list-vpn_1)# **default-action accept**

5. Apply the policy and the cflowd template to sites in the overlay network:
   vSmart(config)# **apply-policy site-list cflowd-sites data-policy test-cflowd-policy**
   vSmart(config-site-list-cflowd-sites)# **cflowd-template test-cflowd-template**

6. Activate the data policy:
   vSmart(config-site-list-cflowd-sites)# **validate**
   Validation complete
   vSmart(config-site-list-cflowd-sites)# **commit**
   Commit complete.
   vSmart(config-site-list-cflowd-sites)# **exit configuration-mode**
   vSmart#

Full Example Configuration

Here is what the full example cflowd configuration looks like:

```
vSmart(config)# show configuration
apply-policy
 site-list cflowd-sites
  data-policy     test-cflowd-policy
  cflowd-template test-cflowd-template
 !
!
policy
 data-policy test-cflowd-policy
  vpn-list vpn_1
   sequence 1
    match
     protocol 6
    !
    action accept
     cflowd
    !
   !
   default-action accept
  !
 !
 cflowd-template test-cflowd-template
  flow-inactive-timeout 60
  template-refresh      90
  collector vpn 1 address 172.16.155.15 port 13322 transport transport_udp
 !
 lists
  vpn-list vpn_1
   vpn 1
  !
  site-list cflowd-sites
   site-id 400,500,600
  !
 !
!
```

Check the Cflowd Configuration

After you activate the cflowd configuration on the vSmart controller, you can check it with the **show running-config policy** and **show running-config apply-policy** commands on the vSmart controller. In addition, the configuration is immediately pushed down to the vEdge routers at the affected sites. You can view the pushed cflowd template with the **show policy from-vsmart cflowd** command. Here is the output from a router at site 500:

```
vEdge# show policy from-vsmart cflowd-template
from-vsmart cflowd-template test-cflowd-template
 flow-active-timeout   30
 flow-inactive-timeout 60
 template-refresh      90
 collector vpn 1 address 172.16.155.15 port 13322 transport transport_udp
```

You can view all the pushed policy components with the **show policy from-vsmart** command:

```
vEdge# show policy from-vsmart
from-vsmart data-policy test-cflowd-policy
 vpn-list vpn_1
  sequence 1
```

```
   match
    protocol 6
   action accept
    cflowd
  default-action accept
from-vsmart cflowd-template test-cflowd-template
 flow-active-timeout   30
 flow-inactive-timeout 60
 template-refresh      90
 collector vpn 1 address 172.16.155.15 port 13322 transport transport_udp
from-vsmart lists vpn-list vpn_1
 vpn 1
```

Check the Flows

On the vEdge routers affected by the cflowd data policy, various commands let you check the status of the cflowd flows.

To display information about the flows themselves:

```
vEdge# show app cflowd flows
                                                                    TCP
TIME
                                    SRC     DEST          IP      CNTRL   ICMP              EGRESS   INGRESS
TOTAL   TOTAL   MIN   MAX                                  TO
VPN   SRC IP         DEST IP        PORT    PORT    DSCP   PROTO   BITS    OPCODE   NHOP IP   INTF     INTF      PKTS
BYTES   LEN   LEN   START TIME                EXPIRE
-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------
1     10.20.24.15   172.16.155.15   46772   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:31:45 2014  3
1     10.20.24.15   172.16.155.15   46773   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:31:50 2014  8
1     10.20.24.15   172.16.155.15   46774   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:31:55 2014  13
1     10.20.24.15   172.16.155.15   46775   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:00 2014  18
1     10.20.24.15   172.16.155.15   46776   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:05 2014  23
1     10.20.24.15   172.16.155.15   46777   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:10 2014  28
1     10.20.24.15   172.16.155.15   46778   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:15 2014  33
1     10.20.24.15   172.16.155.15   46779   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:19 2014  38
1     10.20.24.15   172.16.155.15   46780   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:25 2014  43
1     10.20.24.15   172.16.155.15   46781   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:30 2014  48
1     10.20.24.15   172.16.155.15   46782   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:35 2014  53
1     10.20.24.15   172.16.155.15   46783   13322   0      6       2       0        0.0.0.0   0        0         1
78      78    78    Wed Nov 19 12:32:40 2014  58
```

To quickly get a count of the number of flows:

```
vEdge# show app cflowd flow-count

VPN  count
------------
1    12
```

To display flow statistics:

```
vEdge# show app cflowd statistics

      data_packets             :      0
      template_packets         :      0
      total-packets            :      0
      flow-refresh             :      123
      flow-ageout              :      117
      flow-end-detected        :      0
      flow-end-forced          :      0
```

The following commands show information about the cflowd collectors and the cflowd template information that is sent to the collector:

```
vEdge# show app cflowd collector

VPN  COLLECTOR IP   COLLECTOR  CONNECTION            IPFIX    CONNECTION  TEMPLATE  DATA
ID   ADDRESS        PORT       STATE       PROTOCOL  VERSION  RETRY       PACKETS   PACKETS
-------------------------------------------------------------------------------------------
1    172.16.155.15  13322      false       TCP       10       133         0         0

vEdge# show app cflowd template
app cflowd template name test-cflowd-template
app cflowd template flow-active-timeout 30
app cflowd template flow-inactive-timeout 60
app cflowd template template-refresh 90
```

Additional Information

Configuring Cflowd Traffic Flow Monitoring
Traffic Monitoring with Cflowd

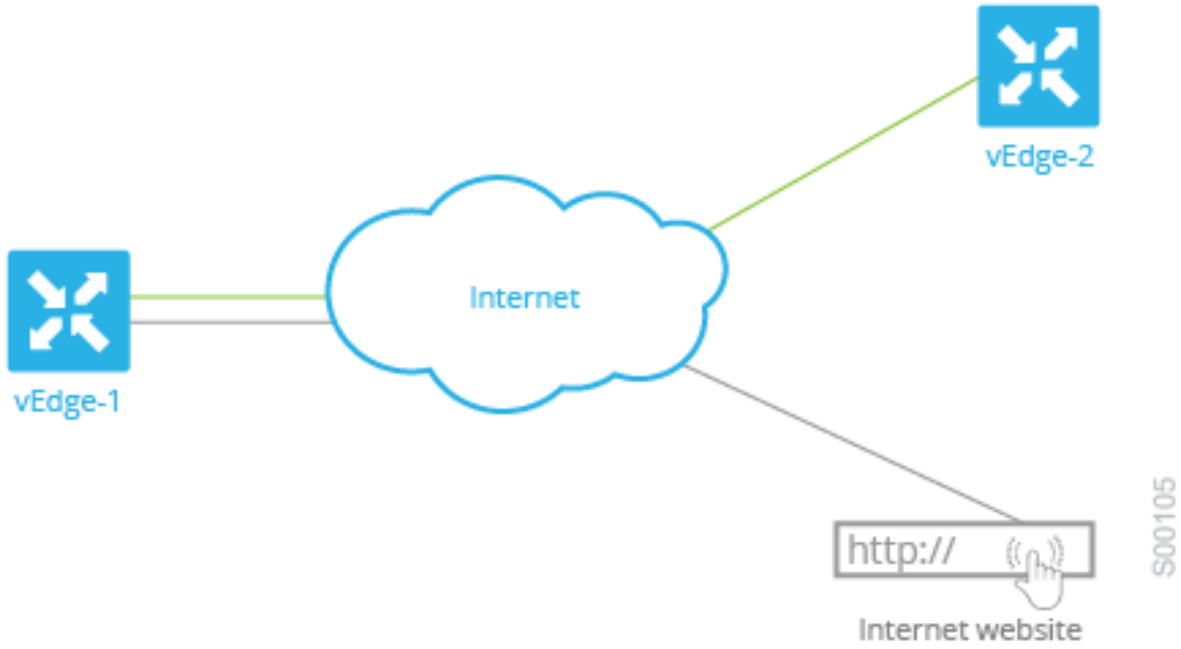
Using a vEdge Router as a NAT Device

A vEdge router can act as a NAT device, both on the transport side of the router and on the service side. On the transport side, the NAT functionality allows traffic from a local site to flow directly to the Internet rather than being backhauled to a colo facility that provides NAT services for Internet access. The NAT function is performed as the traffic enters the overlay tunnel to the WAN transport. On the service side, NAT functionality allows traffic from the local site to traverse the NAT before entering the overlay tunnel.

Using a vEdge Router as a NAT Device on the Transport Side

To provide users at a local site with direct, secure access to Internet resources, such as websites, you can configure the vEdge router to function as a Network Address Translation (NAT) device, performing both address and port translation (NAPT). Enabling NAT allows traffic exiting from a vEdge router to pass directly to the Internet rather than being backhauled to a colocation facility that provides NAT services for Internet access. Using NAT in this way on a vEdge router can eliminate traffic "tromboning" and allows for efficient routes, that have shorter distances, between users at the local site and the network-based applications that they use.

The figure below shows the router acting as a NAT device. The vEdge router splits its traffic into two flows, which you can think of as two separate tunnels. One traffic flow, shown in green, remains within the overlay network and travels between the two routers in the usual fashion, on the secure IPsec tunnels that form the overlay network. The second traffic stream, shown in grey, is redirected through the vEdge router's NAT device and then out of the overlay network to a public network.

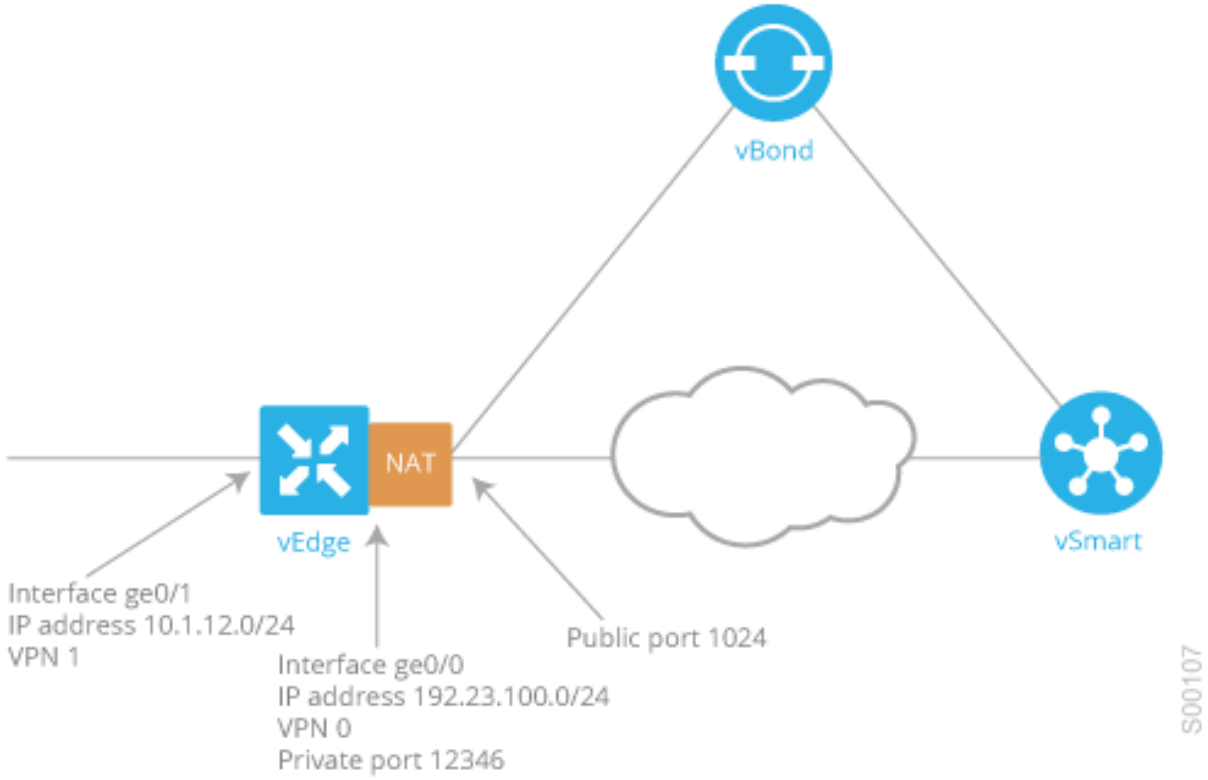


The NAT functionality on a vEdge routers operates in a standard end-point independent fashion. The NAT software performs both address and port translation (NAPT). It establishes a translation entry between a private address and port pair inside the overlay network and a public address and port outside the overlay network. Once this translation entry is created, the NAT software allows incoming connections from an external host to be established with that private address and port only if that private address and port already established a connection to the external host. That is, an external host can reply to traffic from the private address and port; it cannot initiate a connection.

The Viptela NAT software supports 64,000 NAT flows.

Transport-Side NAT Operation

We use the following figure to explain how the NAT functionality on the vEdge router splits traffic into two flows (or two tunnels) so that some of it remains within the overlay network and some goes directly to the Internet or other public network.



In this figure, the vEdge router has two interfaces:

- Interface ge0/1 faces the local site and is in VPN 1. Its IP address is 10.1.12.0/24.
- Interface ge0/0 faces the transport cloud and is in VPN 0 (the transport VPN). Its IP address is 192.23.100.0/24, and it uses the default OMP port number, 12346, for overlay network tunnels.

To configure the vEdge router to act as a NAT device so that some traffic from the router can go directly to a public network, you do three things:

- Enable NAT in the transport VPN (VPN 0) on the WAN-transport–facing interface, which here is ge0/0. All traffic exiting from the vEdge router, going either to other overlay network sites or to a public network, passes through this interface.
- To direct data traffic from other VPNs to exit from the vEdge router directly to a public network, enable NAT in those VPNs or ensure that those VPNs have a route to VPN 0.
- On the vSmart controller, create a centralized data policy the redirects the desired data traffic from the non-transport VPN to VPN 0, and then apply that data policy to the non-transport VPN. In this case, we apply the policy to VPN 1.

Once NAT is enabled on the vEdge router, data traffic affected by the centralized data policy (here, the data traffic from VPN 1) is split into two flows:

- Traffic destined for another vEdge router in the overlay network remains in VPN 1, and it travels directly through the IPsec data plane tunnel from the source vEdge router to the destination vEdge router. This traffic never passes through VPN 0, and therefore it is never touched by NAT.
- Traffic destined for the public network passes from VPN 1 to VPN 0, where it is NATed. During the NAT processing, the source IP address is changed from 10.1.12.0/24 to that of ge0/0, 179.23.100.0/24, and the source port is changed to 1024.

When NAT is enabled, all traffic that passes through VPN 0 is NATed. This includes both the data traffic from VPN 1 that is destined for a public network, and all control traffic, including the traffic required to establish and maintain DTLS control plane tunnels between the vEdge router and the vSmart controller and between the router and the vBond orchestrator.

The vBond orchestrator learns both the public and private addresses of the vEdge router, and it advertises both address to the vSmart controller. In turn, the vSmart controller advertises both addresses to all the vEdge routers in its domain. Each vEdge router then decides whether to use the public or the private address to communicate with another vEdge router as follows:

- If the vEdge router is located at the same site as the other router (that is, if they are both configured with the same overlay network site ID), it communicates using the private address. Because both routers have the same site ID, they are behind the same NAT, and so their communication channels are already secure.
- If the vEdge route is at a different site, it communicates with the other router using the public address. Then, the NAT functionality on the vEdge router translates the public address to the proper private address.

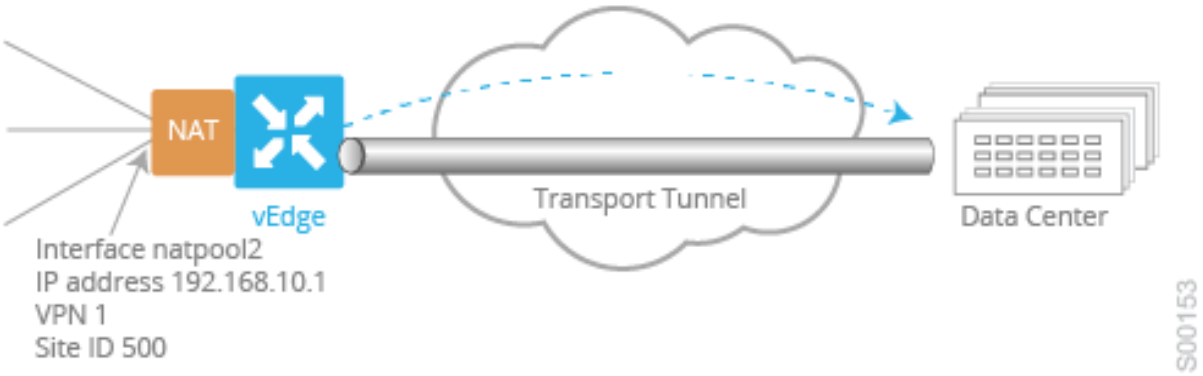
If a vSmart controller connected to a corporate NAT and a NAT-enabled vEdge router are located at the same physical overlay network site, you must configure them with different Viptela site identifiers in order for them to be able to communicate. Similarly, if more than one NAT-enabled vEdge router is located at the same physical overlay network site, each one must be configured with a different site identifier.

Using a vEdge Router as a Service-Side NAT Device

On a vEdge router, you can configure NAT on the service side of the router so that data traffic traverses the NAT before entering the overlay tunnel that is located in the transport VPN. The service-side NAT performs NAT to mask the IP address of data traffic it receives. You can configure both dynamic NAT and 1:1 static NAT on the vEdge router.

Service-Side NAT Operation

We use the following figure to explain how the vEdge router provides NAT services on the service side:



In this figure, the vEdge router has one NAT interface in VPN 1. This interface pools all service-side traffic destined for the NAT interface. The interface name is natpool2, and its IP address is 192.168.10.1. This IP address is the address each packet's IP address is translated to.

To configure the service-side NAT operation on the vEdge router so that traffic traverses the NAT in VPN 1 before being placed on the transport tunnel towards its destination, you do two things:

- Create a NAT pool interface in VPN 1, the service-side VPN. Here, the NAT pool number is 2.
- To direct data traffic from prefixes within VPN 1 to the service-side NAT, create a centralized data policy on the vSmart controller. In the match condition, specify the prefixes to be NATed. In the action condition, set the desired NAT pool, here, natpool 2. Then apply the data policy to the desired site (here, site 500), and apply it to traffic coming from the service side.

When service-side NAT is enabled, all matching prefixes in VPN 1 are directed to the natpool2 interface. This traffic is NATed, with the NAT swapping out the service-side IP address and replacing it with its NAT pool IP address. The packet then gets forwarded to its destination, here the data center.

Additional Information

Configuring Local Internet Exit
Configuring NAT Port Forwarding
Configuring Service-Side NAT

Configuring Local Internet Exit

To configure a vEdge router to be an Internet exit point, you enable NAT within a VPN on the vEdge router, and then you configure a centralized data policy on a vSmart controller. This policy splits the traffic within the VPN so that some of it is directed towards remote sites within the VPN, and hence remains within the overlay network, and other traffic is directed to the Internet or other destinations outside the overlay network. It is also possible to configure a vEdge router to forward data traffic directly to the Internet, by specifying the destination IP prefix.

NAT Configuration Considerations

When configuring a vEdge router to act as a NAT device, keep the following considerations in mind:

- For a vEdge router that is acting as a vBond orchestrator, do not enable NAT operation on the interface that is tied to the vBond orchestrator's IP address. If you do so, the orchestrator is placed into a private address space behind the NAT. For the overlay network to function properly, the vBond orchestrator must be in a public address space. You can, however, enable NAT operation on other vEdge router interfaces.
- When you enable NAT on a vEdge router, the router NATs all traffic that is sent out through VPN 0. That is, both data traffic and control traffic are NATed.
- The NAT operation on outgoing traffic is performed in VPN 0, which is always only a transport VPN. The router's connection to the Internet is in VPN 0. Performing the NAT operation in VPN 0 avoids the IPsec tunnels that carry data traffic within the overlay network.
- If you configure NAT on multiple interfaces in VPN 0, ECMP is performed among the interfaces.
- When you use NAT—either by configuring it on an interface or by setting it as an action in a centralized data policy—no route lookup is performed. Instead, traffic is forwarded to one of the available NAT default gateways.
- The vEdge router NAT implementation uses end-point–independent NAT. If your network contains other NAT devices that interact with the vEdge router NAT, these devices must either perform end-point–independent NAT, or they must be configured with policy rules so that they do not change the port numbers for Viptela overlay network destinations.
- When a vEdge router has two or more NAT interfaces, and hence two or more DIA connections to the internet, by default, data traffic is forwarding on the NAT interfaces using ECMP. To direct data traffic to a specific DIA interface, configure a centralized data policy on the vSmart controller that sets two actions— **nat** and **local-tloc** color. In the **local-tloc** color action, specify the color of the TLOC that connects to the desired DIA connection.

Direct Traffic to Exit to the Internet Using Data Policy

To use a centralized data policy to direct traffic from a vEdge router directly to the Internet, you enable NAT functionality in the WAN VPN or VPNs, and then you create and apply a centralized data policy.

Enable NAT Functionality in the WAN VPN

The first step in setting up Internet exit on a vEdge router is to configure the router to act as a NAT device. You do this by enabling NAT functionality in VPNs that have interfaces that connect to a WAN transport network. By default, VPN 0 always connects to the WAN transport. Other VPNs in your network might also connect to WANs.

To configure a vEdge router to act as a NAT device:

1. Enable NAT in the desired VPN:
   vEdge(config)# **vpn** *vpn-id* **interface** *interface-name* **nat**

2. By default, NAT mappings from the Viptela overlay network side of the NAT to the external side of the NAT remain active, and NAT mapping timers are refreshed regularly to keep the mapping operational. To also refresh NAT mappings of packets coming from the external side of the NAT into the overlay network, change the refresh behavior:
   vEdge(config-nat)# **refresh** **bi-directional**

3. NAT sessions time out after a period of non-use. By default, TCP sessions time out after 60 minutes, and UDP sessions time out after 20 minutes. To change these times:
   vEdge(config-nat)# **tcp-timeout** *minutes*
   vEdge(config-nat)# **udp-timeout** *minutes*
   The times can be from 1 to 65535 minutes.
   The following NAT session timers are fixed, and you cannot modify them:
   • TCP session timeout if no SYN-ACK response is received—5 seconds
   • TCP session timeout if three-way handshaking is not established—10 seconds
   • TCP session timeout after receiving a FIN/RST packet—30 seconds
   • ICMP timeout—6 seconds
   • Other IP timeout—60 seconds

4. By default, the vEdge router does not receive inbound ICMP error messages. However, NAT uses ICMP to relay error messages across a NAT. To have the router receive the NAT ICMP messages:
   vEdge(config-nat)# **no block-icmp-error**
   In case of a DDoS attack, you might want to return to the default, to again prevent the vEdge router from receiving inbound ICMP error messages.

Create a Data Policy to Direct Traffic to the Internet Exit

To direct data traffic from a vEdge router to an Internet exit point, you split the destination of the traffic within a VPN, sending some to remote sites in the VPN and directing the traffic that is destined to the Internet (or other destinations outside the overlay network) to exit directly from the local vEdge router to the external destination.

To split the traffic, configure a centralized data policy on a vSmart controller:

1. Configure the source prefix of the data traffic:
   vSmart(config)# **policy data-policy** *policy-name*
   vSmart(data-policy)# **vpn-list** *list-name*
   vSmart(vpn-list)# **sequence** *number*
   vSmart(sequence)# **match source-ip** *ip-prefix*

2. Configure the destination of the data traffic, either by IP prefix or by port number:
   vSmart(sequence)# **match destination-ip** *ip-prefix*
   vSmart(sequence)# **match destination-port** *port-number*

3. Direct matching data traffic to the NAT functionality. You can optionally configure a packet counter.
   vSmart(sequence)# **action** **accept**
   vSmart(accept)# **count** *counter-name*
   vSmart(accept)# **nat use-vpn 0**

4. Configure additional sequences, as needed, for other source prefixes and destination prefixes or ports, and for other VPNs.

5. Change the default data policy accept default action from reject to accept. With this configuration, all non-matching data traffic is forwarded to service-side VPNs at remote sites instead of being dropped.
   vSmart(vpn-list)# **default-action accept**

6. Apply the data policy to particular sites in the overlay network:
   vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* **from-service**

Direct Traffic To Exit to the Internet Based Only on IP Prefix

You can direct lcal data traffic to exit to the internet based only on the destination IP prefix. To configure this, in the service VPN, forward traffic that is destined towards an internet location to VPN 0, which is the WAN transport VPN:

```
vEdge(config)# vpn vpn-id
vEdge(config-vpn)# ip route prefix vpn 0
```

In the **vpn** command, specify the VPN ID of the service-side VPN from which you are sending the traffic. In the **ip route** command, *prefix* is the IPv4 prefix of the remote destination.The **vpn 0** option configures the software to perform the route lookup in VPN 0 rather than in the service-side VPN. This is done because the service-side VPN cannot resolve the route.

For the traffic redirection to work, in VPN 0, you must enable NAT on the interface associated with the configured prefix:

```
vEdge(config)# vpn 0 interface interface-name nat
```

Here, the interface is the one to use to reach the destination prefix.

The following snippet illustrates the two parts of the configuration:

```
vEdge# show running-config vpn 1
vpn 1
...
 ip route 10.1.17.15/32 vpn 0
!
vEdge# show running-config vpn 0
vpn 0
...
 interface ge0/1
...
  nat
  !
  no shutdown
 !
!
```

To verify that the redirection is working properly, look at the output of the **show ip routes** command:

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

                                    PROTOCOL  NEXTHOP  NEXTHOP          NEXTHOP
VPN  PREFIX              PROTOCOL   SUB TYPE  IF NAME  ADDR            VPN      TLOC IP          COLOR
ENCAP  STATUS
-------------------------------------------------------------------------------------------------------
-------------------------
0    0.0.0.0/0           static     -         ge0/0    10.1.15.13       -        -                -
-      F,S
```

```
0    10.0.20.0/24        connected  -         ge0/3    -               -        -               -
-      F,S
0    10.0.100.0/24       connected  -         ge0/7    -               -        -               -
-      F,S
0    10.1.15.0/24        connected  -         ge0/0    -               -        -               -
-      F,S
0    10.1.17.0/24        connected  -         ge0/1    -               -        -               -
-      F,S
0    57.0.1.0/24         connected  -         ge0/6    -               -        -               -
-      F,S
0    172.16.255.15/32    connected  -         system   -               -        -               -
-      F,S
1    10.1.17.15/32       nat        -         ge0/1    -               0        -               -
-      F,S
1    10.20.24.0/24       ospf       -         ge0/4    -               -        -               -
-      -
1    10.20.24.0/24       connected  -         ge0/4    -               -        -               -
-      F,S
1    10.20.25.0/24       omp        -         -        -               -        172.16.255.16   lte
ipsec  F,S
1    56.0.1.0/24         connected  -         ge0/5    -               -        -               -
-      F,S
1    60.0.1.0/24         omp        -         -        -               -        172.16.255.16   lte
ipsec  F,S
1    61.0.1.0/24         omp        -         -        -               -        172.16.255.16   lte
ipsec  F,S
512  10.0.1.0/24         connected  -         eth0     -               -        -               -
-      F,S
```

In VPN 1, the prefix 10.1.17.15/32 is associated with the protocol "nat", which reflects the configuration of the **ip route** command in VPN 1. For this prefix, the next-hop interface is **ge0/1** , and the next-hop VPN is VPN 0. This prefix is installed into the route table only if the resolving next hop is over an interface on which NAT is enabled.

The prefix that you configure in the **ip route** represents a route in the specified VPN (the service VPN whose ID you enter in the first command above). To direct traffic to that prefix, you can redistribute it into BGP or OSPF:

```
vEdge(config-vpn)# bgp address-family address-family redistribute nat
vEdge(config-vpn)# ospf redistribute nat
```

Track Transport Interface Status

When you enable NAT on a transport interface to allow the local router to forward traffic directly to the internet rather than first forwarding the traffic to a data center router connected to the internet, the router directs data traffic according to the centralized data policy that is applied to that interface, forwarding some traffic directly to the internet (or other external network) and other traffic to other VPNs in the overlay network, including the data center. If the internet or external network becomes unavailable, for example, due to a soft failure, the router has no way to learn of this disruption, and it continues to forward traffic based on the policy rules. The result is that traffic that is being forwarded to the internet is silently dropped.

To prevent the internet-bound traffic from being dropped, you can configure the router to track the status of the transport interface and to redirect the traffic to the non-NATed tunnel on the transport interface when the local internet is unavailable. With tracking enabled, the router periodically probes the path to the internet to determine whether it is up. When it detects that the path is down, the router withdraws the NAT route to the internet destination, and reroutes the traffic to the non-NATed tunnel on the interface so that another router in the overlay network can forward the traffic to the internet. The local router continues to periodically check the status of the path to the interface. When it detects that the path is again functioning, the router reinstalls the NAT route to the internet.

To track the transport interface status, you create a global interface tracker, and then you apply it to the transport interface on which NAT is enabled.

To create a transport interface tracker:

```
vEdge(config)# system
vEdge(config-system)# tracker tracker-name
```

```
vEdge(config-tracker)# endpoint-dns-name dns-name
vEdge(config-tracker)# endpoint-ip ip-address
vEdge(config-tracker)# interval seconds
vEdge(config-tracker)# multiplier number
vEdge(config-tracker)# threshold milliseconds
```

At a minimum, you must specify the IP address or DNS name of a destination on the internet. This is the destination to which the router sends probes to determine the status of the transport interface. You can configure either one IP address or one DNS name.

By default, a status probe is sent every minute (60 seconds). To modify this value, change the time in the **interval** command to a value from 10 through 600 seconds.

By default, the router waits 300 milliseconds to receive a response from the internet destination. To modify the time to wait for a response, change the time in the **threshold** command to a value from 100 through 1000 milliseconds.

By default, after sending three probes and receiving no responses, the router declares that transport interface is down. To modify the number of retries, change the number in the **multiplier** command to a value from 1 through 10.

You can configure up to eight interface trackers.

To apply a tracker to a transport interface:

```
vEdge(config)# vpn 0
vEdge(vpn)# interface interface-name
vEdge(interface)# tracker tracker-name
```

You can apply only one tracker to an interface.

Additional Information

Centralized Data Policy
Configuring NAT Port Forwarding
Using a vEdge Router as a NAT Device

Configuring NAT Port Forwarding

NAT allows requests coming from the internal (local) network to go out to the external network, but it does not allow request from the external network to come to the internal network. This behavior means that it is impossible for an external device to send a packet to a device on the internal network. It also means that device in the internal network cannot operate as a server with regards to the external network.

To allow requests from the external network to reach internal network devices, you configure the vEdge router that sits at the edge of the internal network to be a NAT gateway that performs NAT port forwarding (also called *port mapping*). With such a configuration, the vEdge router sends all packets received on a particular port from an external network to a specific device on the internal (local) network.

Configure NAT Port Forwarding

To configure NAT port forwarding, define one or more port-forwarding rules to send packets received on a particular port from the external network to an internal server:

```
vEdge(config)# vpn 0
vEdge(config-vpn)# interface geslot/port
vEdge(config-interface)# nat
vEdge(config-nat)# port-forward port-start port-number1 port-end port-number2 proto (tcp | udp) private-
vpn vpn-id private-ip-address ip-address
```

Use the **port-start** and **port-end** options to define the desired TCP or UDP port or range of ports. *port-number1* must be less than or equal to *port-number2* . To apply port forwarding to a single port, specify the same port number for the starting and ending numbers. When

applying port forwarding to a range of ports, the range includes the two port numbers that you specify— *port-number1* and *port-number2* . Packets whose destination port matches the configured port or ports are forwarded to the internal server.

Each rule applies either to TCP or UDP traffic. To match the same ports for both TCP and UDP traffic, configure two rules.

For each rule, specify the private VPN in which the internal server resides and the IP address of the internal server. This VPN is one of the VPN identifiers in the overlay network.

You can create up to 128 rules.

Best Practices for Configuring NAT Port Forwarding

Configuring NAT port forwarding can, in some circumstances, make the vEdge router vulnerable to brute-force attacks. The following configuration snippet illustrates a case where the router could fall victim to an SSH brute-force attack:

```
system
  aaa
    auth-order local
interface ge0/0
  description    Internet
  ip address 192.168.50.28/28
  nat
   no block-icmp-error
   respond-to-ping
   port-forward port-start 22 port-end 22 proto tcp
    private-vpn        0
    private-ip-address 192.168.50.28
   !
  !
  tunnel-interface
   encapsulation ipsec
   color public-internet
  !
  no shutdown
 !
```

This configuration creates a port-forwarding rule for TCP port 22, to accept SSH requests from external devices. By itself, this rule provides no opening for brute-force attacks. (As a side note, enabling SSH on a router interface that is connected to the internet is inherently unsafe.) However, problems can arise because of some of the other commands in this configuration:

- **respond-to-ping** —This command allows the vEdge router to respond to ping requests that are sent from the external network. These ping requests bypass any NAT port-forwarding rules that you have configured. In this configuration, the external network is the Internet, so ping requests can come from anywhere. It is recommended that you do not configure the NAT interface to respond to ping requests. If you need to test reachability, configure this command temporarily and then remove it once the reachability testing is complete.
- **private-vpn 0** —The SSH requests are sent to the WAN transport VPN, VPN 0. A best practice is to forward external traffic to a service-side VPN, that is, to a VPN other than VPN 0 or VPN 512.
- **private-ip-address 192.168.50.28** and **ip address 192.168.50.28/28** —The address of the internal server to which external traffic is being sent is the same as the IP address of the WAN interface. For the private IP address, a best practice is to specify the IP address of a service-side device. If you need to specify a private IP address for one of the interfaces on the vEdge router, do not use an address in the transport VPN (VPN 0). If you need to use an address in VPN 0, do not use an interface that is connected to the Internet.
- **auth-order local** —This configuration provides only for local authentication, using the credentials configured on the vEdge router itself. No RADIUS or TACACS server is used to verify the user's SSH login credentials. While this configuration normally does not expose the router to brute-force attacks, here, in the context of the rest of the configuration, it contributes to the router's vulnerability to attack.

Additional Information

Centralized Data Policy
Using a vEdge Router as a NAT Device

Configuring Service-Side NAT

You can configure both dynamic NAT and 1:1 static NAT on the service side of a router. To do so, you create a NAT pool interface within a service VPN on the router, and then you configure a centralized data policy on the vSmart controller. This policy directs data traffic with the desired prefixes to the service-side NAT. Finally, you configure either dynamic NAT or static NAT on the desired NAT pool interfaces.

Create a NAT Pool Interface

On the router, you create a NAT pool interface. This interface NATs data traffic that is directed to it and then forwards the traffic towards its destination.

To create a NAT pool interface:

1. In the desired VPN, create the NAT pool interface:
   vEdge(config-vpn)# **interface** **natpool** *number*
   The pool can have a number from 1 through 31. You refer to this NAT pool number in the action portion of the centralized data policy that you configure to direct data traffic to the pool. You can configure a maximum of 31 NAT pool interfaces in a VPN.

2. Configure the NAT pool interface's IP address:
   vEdge(config-natpool)# **ip address** *prefix* **/** *length*
   The length of the IP address determines the number of addresses that the router can NAT at the same time. For each NAT pool interface, you can configure a maximum of 250 IP addresses.

3. Enable the interface:
   vEdge(config-natpool)# **no shutdown**

On a NAT pool interface, you can configure only these two commands (**ip address** and **shutdown** / **no shutdown**) and the **nat** command, discussed below. You cannot configure any of the other interface commands.

Here is an example of configuring the NAT pool interface:

```
vEdge# show running-config vpn 1
vpn 1
 interface ge0/4
  ip address 10.20.24.15/24
  no shutdown
 !
 interface ge0/5
  ip address 56.0.1.15/24
  no shutdown
 !
 interface natpool2
  ip address 192.179.10.1/32
  nat
  !
  no shutdown
 !
!
```

To display information about the NAT pool interface, use the **show interface** command:

```
vEdge# show interface vpn 1

                                  IF      IF
TCP
                                  ADMIN   OPER    ENCAP  PORT                                SPEED
```

```
MSS                  RX       TX
VPN  INTERFACE  IP ADDRESS       STATUS  STATUS  TYPE   TYPE     MTU   HWADDR             MBPS   DUPLEX
ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
-------------------------------------
1    ge0/4      10.20.24.15/24   Up      Up      null   service  1500  00:0c:29:7d:1e:26  10     full
1420    0:01:24:06  566      565
1    ge0/5      56.0.1.15/24     Up      Up      null   service  1500  00:0c:29:7d:1e:30  10     full
1420    0:01:24:06  26       4
1    natpool2   192.179.10.1/32  Up      Up      null   service  1500  00:00:00:00:00:00  10     full
1420    0:00:40:57  0        0
```

Create a Data Policy To Direct Data Traffic to a Service-Side NAT

To direct data traffic from the service side of the router to the NAT, you create a centralized data policy on the vSmart controller. In the match condition of the policy, you identify the data traffic that you want to direct to the NAT. One way to do this is to match on the IP prefixes of the data traffic. In the action condition of the policy, you direct the matching traffic to one of the number NAT pools. Finally, you apply the policy to the service side at the desired overlay network sites.

To create a data policy to direct data traffic to a service-side NAT:

1. Configure the lists required for the data policy. You must configure a list of VPN and sites. If you are matching on data prefixes, configure a data prefix list.
   vSmart(config-policy-lists)# **vpn-list** *list-name*
   vSmart(config-policy-vpn-list)# **vpn** *vpn-id*

   vSmart(config-policy-lists)# **site-list** *list-name*
   vSmart(config-policy-site-list)# **site-id** *site-id*

   vSmart(config-policy-lists)# **data-prefix-list** *list-name*
   vSmart(config-policy-data-prefix-list)# **ip-prefix** *prefix* **/** *length*

2. Configure a data policy:
   vSmart(config-policy)# **data-policy** *policy-name*
   vSmart(config-data-policy)# **vpn-list** *list-name*
   vSmart(config-vpn-list)# **sequence** *number*

3. Configure the desired match conditions:
   vSmart(config-sequence)# **match** *condition*

4. In the action, associate matching data traffic with the desired NAT pool:
   vSmart(config-sequence)# **action accept**
   vSmart(config-sequence)# **nat pool** *number*

5. Configure the desired default action for the data policy:
   vSmart(config-vpn-list)# **default-action** (**accept** | **reject**)

6. Apply the policy to the desired sites in the overlay network:
   vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* **from-service**

Here is an example of configuring the centralized data policy:

```
vSmart# show running-config policy
policy
 data-policy service-side-nat-policy
  vpn-list vpn-1
   sequence 10
    match
     source-data-prefix-list prefixes-to-nat
    !
    action accept
```

```
      nat pool 2
     !
    !
    default-action accept
   !
  !
  lists
   vpn-list vpn-1
    vpn 1
   !
   data-prefix-list prefixes-to-nat
    ip-prefix 56.0.1.0/24
   !
   site-list site-500
    site-id 500
   !
  !
!
vSmart# show running-config apply-policy
apply-policy
 site-list site-500
  data-policy service-side-nat-policy from-service
 !
!
```

After you activate the policy, you can see that it has been applied to the router:

```
vEdge# show policy from-vsmart
from-vsmart data-policy service-side-nat-policy
 direction from-service
 vpn-list vpn-1
  sequence 10
   match
    source-data-prefix-list prefixes-to-nat
   action accept
    nat pool 2
  default-action accept
from-vsmart lists vpn-list vpn-1
 vpn 1
from-vsmart lists data-prefix-list prefixes-to-nat
 ip-prefix 56.0.1.0/24
```

Configure Dynamic NAT

By default, when you configure a router to act as a NAT, the router performs dynamic network address translation. In this capacity, the router can perform dynamic NAT for up to 250 IP addresses across NAT pools.

To configure dynamic NAT:

1. In the desired VPN, create the NAT pool interface:
   vEdge(config-vpn)# **interface** **natpool** *number*
   The pool can have a number from 1 through 31. You refer to this NAT pool number in the action portion of the centralized data policy that you configure to direct data traffic to the pool. You can configure a maximum of 31 NAT pool interfaces in a VPN.

2. Configure the IP address prefix for the NAT pool interface:
   vEdge(config-natpool)# **ip address** *prefix* **/** *length*
   The prefix length determines the maximum number of addresses that the router can NAT at the same time. For example, for a /30 prefix length, the router can perform translation on four addresses at a time. For each NAT pool interface, you can configure a maximum of 250 IP addresses.

3. Enable the interface:
   vEdge(config-natpool)# **no shutdown**

4. Enable dynamic NAT:
   vEdge(config-natpool)# **nat**

As mentioned above, the length of the IP address determines the number of IP addresses that the router can NAT at the same time, up to a maximum of 250 across all NAT pools. When all available IP addresses have been used, the router reuses the last IP address multiple times, changing the port number. The port number is chosen at random from the nonreserved port numbers, that is, those port numbers in the range 1024 through 65535. For example, if the IP address is 10.1.17.3/30, the vEdge router can uniquely NAT four IP addresses. Let's say that the router maps the fourth IP address to 10.1.20.5, or more specifically to 10.1.20.5:12346 if we include the port number. It would then map the fifth IP address to the same IP address, but with a different port, such as 10.1.20.5:12347. To have the router drop packets when no more IP addresses are available for the translation process, include the following command:

```
vEdge(config)# vpn vpn-id interface natpoolnumber
vEdge(config-natpool)# no overload
```

Configure Static NAT

You can configure a router acting as a NAT to perform static network address translation (also called 1:1 static NAT) of source IP addresses. You can translate service-side source addresses before sending packets out to the overlay network, and you can translate external addresses before forwarding packets to the service-side network. You can also translate service-side source addresses before sending packets out to another service-side LAN connected to the same router.

For packets originating on the service side of a router, you can statically map the packets' source IP address to another IP address. You do this by creating a NAT pool interface within a service-side VPN. For this interface, you configure a pool of IP addresses to use for network address translation, and then you configure the static address mappings. When the address pool is depleted, you can choose to drop packets that have unmapped source IP addresses. (Dropping these packets is not the default behavior.)

For packets exiting a transport tunnel from a router, you can statically map the packet's source IP address to another IP address, generally to an address that is routable within the service-side network. You configure this in the same way as for NATing packets originating on the service side.

You must create separate NAT pool interfaces to translate source IP addresses for service-side packets and for tunnel packets.

Across all NAT pools, a vEdge router can NAT a maximum of 254 source IP addresses. This is the number of addresses in a /24 prefix, less the.0 and.255 addresses. You cannot configure translation for.0 and.255 addresses.

This section explains how to configure static NAT for translating service-side source IP addresses and for translating external (transport-side) IP addresses. The two procedures are very similar, but we describe them separately for clarity.

Static NATing of Service-Side Addresses

To configure the static NATing of service-side source IP addresses:

1. In the desired VPN, create the NAT pool interface:
   vEdge(config-vpn)# **interface** **natpool** *number*
   The pool can have a number from 1 through 31. You refer to this NAT pool number in the action portion of the centralized data policy that you configure to direct data traffic to the pool. You can configure a maximum of 31 NAT pool interfaces in a VPN.

2. Enable the NAT pool interface:
   vEdge(config-natpool)# **no shutdown**

3. Configure the IP address prefix for the NAT pool interface:
   vEdge(config-natpool)# **ip address** *prefix* **/** *length*
   The prefix length determines the maximum number of source IP addresses that can be NATed in the NAT pool. For example, for a /30 prefix length, a maximum of four source IP addresses can be NATed. For each NAT pool interface, you can configure a maximum of 250 IP addresses.

4. Configure the NAT pool interface to perform network address translation:
   vEdge(config-natpool)# **nat**

5. By default, all IP addresses are translated to an address in the pool of NAT addresses configured in the **ip address** command. The addresses are mapped one to one until the address pool is depleted. Then, the first address is used multiple times, and the port number is changed to a random value between 1024 and 65535. This reuse of the last address is called *overloading* . Overloading effectively implements dynamic NAT.
   To configure static NAT, include the **no overload** command to enforce the mapping of a single source IP address to a single translated IP address:
   vEdge(config-nat)# **no overload**
   With this command, when the maximum number of available IP addresses available to be translated is reached, packets with other IP addresses are dropped.

6. Set the direction in which the NAT pool interface performs static mapping to **inside** to statically translate service-side IP source addresses:
   vEdge(config-nat)# **direction inside**
   Note that the default direction is **inside** .
   A single NAT pool interface can perform static address translation either for service-side source addresses (**direction inside**) or for external source addresses (**direction outside**), but not for both. This means that for a single NAT pool, you can configure only one **direction** command.

7. Define the static address translations for service-side source IP addresses:
   vEdge(config-nat)# **static source-ip** *ip-address1* **translate-ip** *ip-address2* **inside**
   ip-address1 is the source IP address of a device or branch router on the service side of the vEdge router.
   ip-address2 is the translated source IP address. This is the address that the vEdge router places in the source field of the packet's IP header when transmitting the packet out the transport network. Because the NAT pool direction is **inside** , this IP address must be in the interface's IP address range. This is the IP address prefix configured in the **ip address** command.
   The **inside** option indicates that it is a service-side, or inside, address that is being statically translated. Note that the **inside** option in the **static** command is different from and independent of the **inside** or **outside** option you specify in the **direction** command. When you are statically NATing service-side addresses, you can statically map both service-side addresses (with a **static** ... **inside** command) and transport-side addresses (with a **static** ... **outside** command), as described in the next step.
   The maximum number of service-side source IP addresses that you can statically NAT is equal to the number of addresses available in the interface's prefix range. For example, for a /30 prefix length, you can configure a maximum of four static NAT mappings.
   Once the NAT static address mapping is installed in the router's NAT table, the router can perform source IP address translation in both directions—when a service-side packet is being transmitted into the transport network, and when an external packet (addressed to *ip-address2*) arrives at the router.

8. Define the static address translations for transport-side source IP addresses:
   vEdge(config-nat)# **static source-ip** *ip-address1* **translate-ip** *ip-address2* **outside**
   ip-address1 is the source IP address of an external device or router, that is, of a device at a remote site.
   ip-address2 is the translated source IP address. This is the address that the vEdge router places in the source field of the packet's IP header before forwarding the traffic to the service-side network.
   The **outside** option indicates that an external IP address is being statically translated. Note that the **outside** option in the **static** command is different from and independent of the **inside** or **outside** option you specify in the **direction** command. When you are statically NATing service-side addresses, you can statically map both service-side addresses (with a **static** ... **inside** command) and transport-side addresses (with a **static** ... **outside** command), as described in the previous step.
   Because the direction of the NAT pool is **inside** , the pool of IP addresses set aside for NATing is used only to NAT service-side source IP addresses. This means that here, you can configure any number of external static address translations.
   As a corollary of NATing an external IP address, when a service-side device responds to that external IP address, it simply takes the source IP address from the received packet and places it into the destination IP field in the IP header.

9. Optionally, log the creation and deletion of NAT flows:
   vEdge(config-nat)# **log-translations**

Static NATing of External Addresses

To configure the static NATing of external source IP addresses:

1. In the desired VPN, create the NAT pool interface:
   vEdge(config-vpn)# **interface natpool** *number*

The pool can have a number from 1 through 31. You refer to this NAT pool number in the action portion of the centralized data policy that you configure to direct data traffic to the pool. You can configure a maximum of 31 NAT pool interfaces in a VPN.

2. Enable the NAT pool interface:
   vEdge(config-natpool)# **no shutdown**

3. Configure the NAT pool interface to perform network address translation:

4. Configure the IP address prefix for the NAT pool interface:
   vEdge(config-natpool)# **ip address** *prefix* **/** *length*
   The prefix length determines the maximum number of IP addresses that the router can NAT at the same time in that NAT pool. For example, for a /30 prefix length, the router can perform translation on four addresses at a time. For each NAT pool interface, you can configure a maximum of 250 IP addresses.

5. Configure the NAT pool interface to perform network address translation:
   vEdge(config-natpool)# **nat**

6. By default, all IP addresses are translated to an address in the pool of NAT addresses configured in the **ip address** command. The addresses are mapped one to one until the address pool is depleted. Then, the last address is used multiple times, and the port number is changed to a random value between 1024 and 65535. This reuse of the last address is called *overloading* . Overloading effectively implements dynamic NAT.
   To configure static NATing of external addresses, you must include the **no overload** command to enforce the mapping of a single source IP address to a single translated IP address, because the software does not support overloading on the outside NAT pool interface:
   vEdge(config-nat)# **no overload**
   With this command, when the maximum number of available IP addresses available to be translated is reached, packets with other IP addresses are dropped.

7. Set the direction in which the NAT pool interface performs static mapping to **outside** to statically translate external IP source addresses:
   vEdge(config-nat)# **direction out side**
   The default direction is **inside** .
   A single NAT pool interface can perform static address translation either for service-side source addresses (**direction inside**) or for external source addresses (**direction outside**), but not for both. This means that for a single NAT pool, you can configure only one **direction** command.

8. Define the static address translations for external source-IP addresses:
   vEdge(config-nat)# **static source-ip** *ip-address1* **translate-ip** *ip-address2* **outside**
   ip-address1 is the source IP address of a remote device or router on the transport side of the router.
   ip-address2 is the translated source IP address. This is the address that the router places in the source field of the packet's IP header when forwarding the packet into the service-side network. Because the NAT pool direction is **outside** , this IP address must be in the interface's IP address range. This is the IP address prefix configured in the **ip address** command.
   The **outside** option indicates that it is an external, or outside, address that is being statically translated. Note that the **outside** option in the **static** command is different from and independent of the **inside** or **outside** option you specify in the **direction** command. When you are statically NATing external addresses, you can statically map both transport-side addresses (with a **static** ... **outside** command) and service-side addresses (with a **static** ... **inside** command), as described in the previous step.
   The maximum number of external source IP addresses that you can statically NAT is equal to the number of addresses available in the interface's prefix range. For example, for a /30 prefix length, you can configure a maximum of four static NAT mappings.
   As a corollary of NATing an external IP address, when a service-side device responds to that external IP address, it simply takes the source IP address from the received packet and places it into the destination IP field in the IP header.

Additional Information

Configuring Centralized Data Policy
Service-Side NAT Configuration Example
Troubleshooting Static NATing of Service-Side NAT
Using a vEdge Router as a NAT Device

Configuring Split DNS

When an application-aware routing policy allows a vEdge router to send application traffic to and receive application traffic from a service VPN, the router performs a Domain Name System (DNS) lookup to determine how to reach a server for the application. If the router does not have a connection to the internet, it sends DNS queries to a router that has such a connection, and that router determines how to reach a server for that application. In a network in which the internet-connect router is in a geographically distant data center, the resolved DNS address might point to a server that is also geographically distant from the site where the service VPN is located.

Because you can configure a vEdge router to be an internet exit point, it is possible for any router to reach the internet directly to perform DNS lookups. To do this, you create a policy that configures split DNS and that defines, on an application-by-application basis, how to perform DNS lookups.

You configure split DNS with either a centralized data policy or, if you want to apply SLA criteria to the data traffic, an application-aware routing policy. You create these policies on a vSmart controller, and they are pushed to the vEdge routers.

CLI Configuration Procedure

Configure Split DNS with a Centralized Data Policy

The following high-level steps show the minimum policy components required to enable split DNS with a centralized data policy:

1. Create one or more lists of overlay network sites to which the centralized data policy is to be applied (in an **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).

2. Create lists of applications or application families for which you want to enable split DNS. You refer to these lists in the **match** section of the data policy.
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **app-list** *list-name*
   vSmart(config-app-list)# (**app** *application-name* | **app-family** *family-name*)

3. Create lists VPNs to which the split DNS policy is to be applied (in a **policy data-policy** command):
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists)# **vpn** *vpn-id*

4. Create a data policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy data-policy** *policy-name*
   vSmart (config-data-policy)# **vpn-list** *list-name*

5. Create a series of match–action pair sequences:
   vSmart(config-vpn-list)# **sequence** *number*
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

6. Process the DNS server resolution for the applications or application families contained in an application list. In *list-name* , specify one of the names in a **policy lists app-list** command.
   vSmart(config-sequence)# **match dns-app-list** *list-name*

7. Configure the match–action pair sequence to process DNS requests (for outbound data traffic) or responses (for inbound data traffic):
   vSmart(config-sequence)# **match dns** (**request** | **response**)

8. Accept matching packets, optionally counting and logging them:
   vSmart(config-sequence)# **action accept** [**count** *counter-name*] [**log**]

9. Enable local internet exit:
   vSmart(config-sequence)# **action accept nat** [**pool** *number*] [**use-vpn 0**]

10. By default, the DNS servers configured in the VPN in which the policy is applied are used to process DNS lookups for the applications. You can direct DNS requests to a particular DNS server. For a data policy condition that applies to outbound traffic (from the service network), configure the IP address of the DNS server:
    vSmart(config-sequence)# **action accept redirect-dns** *ip-address*
    For a data policy condition that applies to inbound traffic (from the tunnel), include the following so that the DNS response can be correctly forwarded back to the service VPN:
    vSmart(config-sequence)# **action accept redirect-dns host**

11. f a route does not match any of the conditions in one of the sequences, it is rejected by default. To accept nonmatching prefixed, configure the default action for the policy:
    vSmart(config- *policy-name*)# **default-action accept**

12. Apply the policy to one or more sites in the overlay network:
    vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* (**all** | **from-service** | **from-tunnel**)

Configure Split DNS with an Application-Aware Routing Policy

The following high-level steps show the minimum policy components required to enable split DNS with an application-aware routing policy:

1. Create one or more lists of overlay network sites to which the centralized data policy is to be applied (in an **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists- *list-name*)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).

2. Create SLA classes and traffic characteristics to apply to matching application data traffic:
   vSmart(config)# **policy sla-class** *sla-class-name*
   vSmart(config-sla-class)# **jitter** *milliseconds*
   vSmart(config-sla-class)# **latency** *milliseconds*
   vSmart(config-sla-class)# **loss** *percentage*

3. Create lists of applications or application families to identify application traffic of interest in the **match** section of the data policy:
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **app-list** *list-name*
   vSmart(config-app-list)# (**app** *application-name* | **app-family** *family-name*)

4. Create lists VPNs to which the split DNS policy is to be applied (in a **policy data-policy** command):
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists- *list-name*)# **vpn** *vpn-id*

5. If you are configuring a logging action, configure how often to log packets to syslog files:
   vEdge(config)# **policy log-frequency** *number*

6. Create an application-aware routing policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy app-route-policy** *policy-name*
   vSmart (config-data-policy- *policy-name*)# **vpn-list** *list-name*

7. Create a series of match–pair sequences:
   vSmart(config-vpn-list)# **sequence** *number*
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route

matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

8. Process the DNS server resolution for the applications or application families contained in an application list. In *list-name* , specify one of the names in a **policy lists app-list** command.
   vSmart(config-sequence- *number*)# **match dns-app-list** *list-name*

9. Configure the match–action pair sequence to process s DNS requests (for outbound data traffic) or responses (for inbound data traffic):
   vSmart(config-sequence- *number*)# **match** (**request** | **response**)

10. Define the SLA action to take if a match occurs:
    vSmart(config-sequence)# **action sla-class** *sla-class-name* [**strict**]
    vSmart(config-sequence)# **action sla-class** *sla-class-name* [**strict**] **preferred-color** *colors*
    vSmart(config-sequence)# **action backup-sla-preferred-color** *colors*

11. For matching packets, optionally count and log them:
    vSmart(config-sequence)# **action count** *counter-name*
    vSmart(config-sequence)# **action log**

12. Enable local internet exit:
    vSmart(config-sequence- *number*)# **action acccept nat** [**pool** *number*] [**use-vpn 0**]

13. If a packet does not match any of the conditions in one of the sequences, a default action is taken. For application-aware routing policy, the default action is to accept nonmatching traffic and forward it with no consideration of SLA. You can configure the default action so that SLA parameters are applied to nonmatching packets:
    vSmart(config- *policy-name*)# **default-action sla-class** *sla-class-name*

14. Apply the policy to one or more sites in the overlay network:
    vSmart(config)# **apply-policy site-list** *list-name* **app-route-policy** *policy-name*

Structural Components of Policy Configuration for Split DNS

Below are the structural components required to configure split DNS on a vSmart controller. The components related to configuring split DNS are explained in the sections below. For an explanation of the data policy and application-aware routing policy components that are not specifically related to split DNS, see Configuring Centralized Data Policy and Configuring Application-Aware Routing .

```
policy
  lists
    app-list list-name
      (app application-name | app-family application-family)
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn-id vpn-id
 data-policy policy-name
   vpn-list list-name
     sequence number
       match
         dns (request | response)
         dns-app-list list-name
       action accept
         count counter-name
         log
         nat use-vpn 0
         redirect-dns (ip-address | host)
     default-action
       (accept | drop)
apply-policy
  site-list list-name data-policy policy-name (all | from-service | from-tunnel)
```

```
policy
  lists
    app-list list-name
      (app application-name | app-family application-family)
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn-id vpn-id
log-frequency number
sla-class sla-class-name
  jitter milliseconds
  latency milliseconds
  loss percentage
app-route-policy policy-name
   vpn-list list-name
     sequence number
       match
         dns (request | response)
         dns-app-list list-name
       action
         backup-sla-preferred-color colors
         count counter-name
         log
         nat use-vpn 0
         sla-class sla-class-name [strict] [preferred-color colors]
     default-action
       sla-class sla-class-name
apply-policy
  site-list list-name app-route-policy policy-name
```

Lists

A data policy or an application-aware routing policy for split DNS uses the following types of lists to group related items. You configure these lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command
Applications and application families	List of one or more applications or application families running on the subnets connected to the vEdge router. Each **app-list** can contain either applications or application families, but you cannot mix the two. To configure multiple applications or application families in a single list, include multiple **app** or **app-family** options, specifying one application or application family in each **app** or **app-family** . • *application-name* is the name of an application. The Viptela software supports about 2300 different applications. To list the supported applications, use the ? in the CLI. • *application-family* is the name of an application family. It can be one of the following: **antivirus** , **application-service** , **audio_video** , **authentication** , **behavioral** , **compression** , **database** , **encrypted** , **erp** , **file-server** , **file-transfer** , **forum** , **game** , **instant-messaging** , **mail** , **microsoft-office** , **middleware** , **network-management** , **network-service** , **peer-to-peer** , **printer** , **routing** , **security-service** , **standard** , **telephony** , **terminal** , **thin-client** , **tunneling** , **wap** , **web** , and **webmail** .	**app-list** *list-name* (**app** *application-name* | **app-family** *application-family*)
Sites	List of one or more site identifiers in the overlay network. To configure multiple sites in a single list, include multiple **site-id** options, specifying one site number in each option. You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*
VPNs	List of one or more VPNs in the overlay network. To configure multiple VPNs in a single list, include multiple **vpn** options, specifying one VPN number in each option. You can specify a single VPN identifier (such as **vpn-id 1**) or a range of VPN identifiers (such as **vpn-id 1-10**).	**vpn-list** *list-name* **vpn** *vpn-id*

In the vSmart controller configuration, you can create multiple iterations of each type of list. For example, it is common to create multiple site lists and multiple VPN lists so that you can apply data policy to different sites and different customer VPNs across the network.

When you create multiple iterations of a type of list (for example, when you create multiple VPN lists), you can include the same values or overlapping values in more than one of these list. You can do this either on purpose, to meet the design needs of your network, or you can

do this accidentally, which might occur when you use ranges to specify values. (You can use ranges to specify data prefixes, site identifiers, and VPNs.) Here are two examples of lists that are configured with ranges and that contain overlapping values:

- **vpn-list list-1 vpn 1-10**
  vpn-list list-2 vpn 6-8

- **site-list list-1 site 1-10**
  site-list list-2 site 5-15

When you configure data policies that contain lists with overlapping values, or when you apply data policies, you must ensure that the lists included in the policies, or included when applying the policies, do not contain overlapping values. To do this, you must manually audit your configurations. The Viptela configuration software performs no validation on the contents of lists, on the data policies themselves, or on how the policies are applied to ensure that there are no overlapping values.

If you configure or apply data policies that contain lists with overlapping values to the same site, one policy is applied and the others are ignored. Which policy is applied is a function of the internal behavior of Viptela software when it processes the configuration. This decision is not under user control, so the outcome is not predictable.

VPN Lists

Each data or application-aware policy instance is associated with a VPN list. You configure VPN lists with the **policy data-policy vpn-list** or **policy app-route-policy vpn-list** command. The VPN list you specify must be one that you created with a **policy lists vpn-list** command.

Sequences

Within each VPN list, a data policy or an application-aware policy contains sequences of match–action pairs. The sequences are numbered to set the order in which data traffic is analyzed by the match–action pairs in the policy. You configure sequences with the **policy data-policy vpn-list sequence** or **policy app-aware-policy vpn-list sequence** command.

Each sequence in a policy can contain one **match** command and one **action** command.

Match Parameters

For a data policy or an application-aware routing policy for split DNS, you must the following two match conditions. You configure the match parameters with the **match** command under the **policy data-policy vpn-list sequence** or **policy app-route-policy vpn-list sequence** command hierarchy on vSmart controllers.

Description	Command	Value or Range
Enable split DNS, to resolve and process DNS requests and responses on an application-by-application basis	**dns-app-list** *list-name*	Name of an **app-list** list. This list specifies the applications whose DNS requests are processed.
Specify the direction in which to process DNS packets	**dns** (**request** | **response**)	To process DNS requests sent by the applications (for outbound DNS queries), specify **dns request** . To process DNS responses returned from DNS servers to the applications, specify **dns response** .

Action Parameters

When data traffic matches the match parameters, the specified action is applied to it. You configure the action parameters with the **action** command under the **policy data-policy vpn-list sequence** or **policy app-route-policy vpn-list sequence** command hierarchy on vSmart controllers.

For application-aware routing policy, the action is to apply an SLA class, which defines the maximum packet latency or maximum packet loss, or both, for DNS traffic related to the application. For information about these action parameters, see Configuring Application-Aware Routing .

For a centralized data policy that enables split DNS, configure the following actions. You can configure other actions, as described in Configuring Centralized Data Policy .

Description	Command	Value or Range
Direct data traffic to an Internet exit point on the local router	**nat use-vpn 0**	—
Count matching data packets. Counting packets is optional, but recommended.	**action count** *counter-name*	Name of a counter.
Redirect DNS requests to a particular DNS server. Redirecting requests is optional, but if you do so, you must specify both actions.	**redirect-dns host** **redirect-dns** *ip-address*	For an inbound policy, **redirect-dns host** allows the DNS response to be correctly forwarded back to the requesting service VPN. For an outbound policy, specify the IP address of the DNS server.

Default Action

If a data packet being evaluated does not match any of the match conditions in a policy, a default action is applied. By default, the data packet is dropped. To modify this behavior, include the **policy data-policy vpn-list default-action accept** command.

Applying a Policy

For an application-aware route policy to take effect, you apply it to a list of sites in the overlay network:

```
vSmart(config)# apply-policy site-list list-name app-route-policy policy-name
```

When you apply the policy, you do not specify a direction (either inbound or outbound). Application-aware routing policy affects only the outbound traffic on the vEdge routers.

For a centralized data policy to take effect, you apply it to a list of sites in the overlay network:

```
vSmart(config)# apply-policy site-list list-name data-policy policy-name (all | from-service | from-tunnel)
```

For split DNS to work, you apply a policy to DNS requests originated from a server VPN. If you are specifying the address of a DNS server for a particular application, the *policy-name* data policy must contain a **redirect-dns** *ip-address* action that applies to that application.

```
vSmart(config)# apply-policy policy-name site-list list-name data-policy policy-name from-service
```

You also apply a policy to DNS responses being returned from the internet. If you included a **redirect-dns** action in the outbound policy, the *policy-name* data policy must contain a **redirect-dns host** action that applies to the proper application.

```
vSmart(config)# apply-policy policy-name site-list list-name data-policy policy-name from-tunnel
```

You can apply the same policy to traffic coming from the service VPN and from the tunnel interface between the router and the internet. If the policy specifies use of a specific DNS for a particular application, the policy must contain two sequences for that application, one with a **request-dns** *ip-address* action and the second with a **request-dns host** action.

```
vSmart(config)# apply-policy policy-name site-list list-name data-policy policy-name all
```

Example Configuration

The following example shows a data policy that enables split DNS for a number of applications and counts the DNS traffic:

```
vSmart# show running-config policy
policy
 data-policy split_dns
  vpn-list vpn_1
   sequence 1
    match
     dns-app-list facebook
     dns          request
    !
    action accept
     count facebook_app
    !
   !
   sequence 2
    match
     dns-app-list concur
     dns          request
    !
    action accept
     count concur-app
     nat use-vpn 0
     redirect-dns 75.0.0.1
    !
   !
   sequence 3
    match
     dns-app-list yahoo
    !
    action accept
     count yahoo-app
     nat use-vpn 0
     redirect-dns 75.0.0.1
    !
   !
   sequence 4
    match
     dns-app-list salesforce
    !
    action accept
     count salesforce
     nat use-vpn 0
     redirect-dns 75.0.0.1
    !
   !
   sequence 5
    match
     dns-app-list twitter
     dns          request
    !
    action accept
     count twitter
     nat use-vpn 0
     redirect-dns 75.0.0.1
    !
   !
   sequence 9
    match
     dns-app-list dns_list
     dns          request
    !
    action accept
     count dns_app_list_count
     nat use-vpn 0
     redirect-dns 75.0.0.1
```

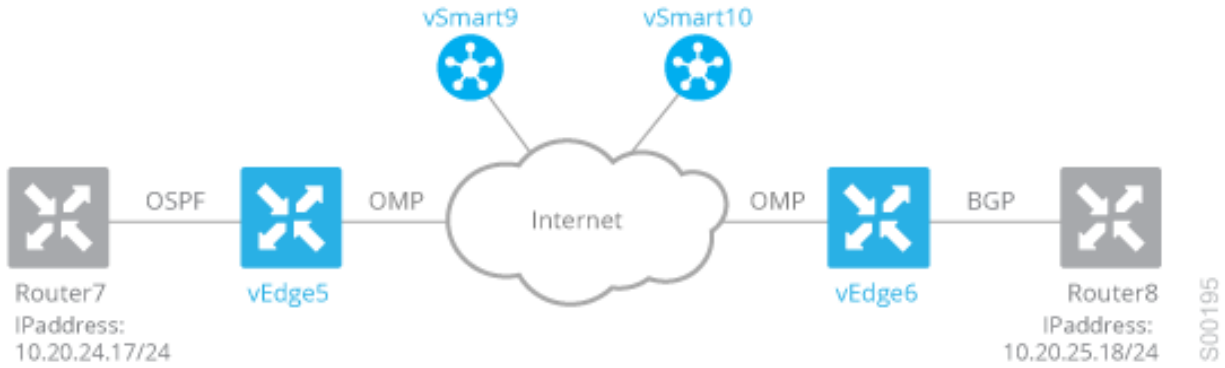
```
      !
     !
     sequence 10
      match
       app-list dns_list
      !
      action accept
       count dns_list_count
       nat use-vpn 0
       redirect-dns 75.0.0.1
      !
     !
     default-action accept
    !
   !
   lists
    vpn-list vpn_1
     vpn 1
    !
    app-list concur
     app concur
    !
    app-list dns_list
     app dns
    !
    app-list facebook
     app facebook
    !
    app-list gmail
     app gmail
     app gmail_basic
     app gmail_chat
     app gmail_drive
     app gmail_mobile
    !
    app-list intuit
     app intuit
    !
    app-list salesforce
     app salesforce
    !
    app-list twitter
     app twitter
    !
    app-list yahoo
     app yahoo
    !
    app-list zendesk
     app zendesk
    !
    site-list vedge1
     site-id 500
    !
   !
  !
  vSmart# show running-config apply-policy
  apply-policy
    site-list vedge1 data-policy split_dns all
```

Additional Information

Configuring Application-Aware Routing
Configuring Centralized Data Policy
Use a vEdge Router as a NAT Device

Service-Side NAT Configuration Example

In this service-side NAT configuration example, two vEdge routers—vEdge5 and vEdge6—are located at two different sites in the overlay network and onnected to each other via the Internet. They are both configured as NATs. Router7 sits in the service side behind vEdge5, and the local network at this site runs OSPF. Router8 sits behind vEdge6 on a network running IBGP.



vEdge5 NATs the source IP address 10.20.24.17, which originates on Router7, translating it to 10.15.1.4. From a NAT perspective on vEdge5, the address 10.20.24.17 is an inside address.

When vEdge6 receives packets with the source IP address 10.15.1.4, it translates the address to 10.16.1.4. From a NAT perspective on vEdge6, the address 10.15.1.4 is an outside address.

In addition, vEdge5 NATs the outside IP source address 10.20.25.18, which originates on Router8 (behind vEdge6), translating it to 10.25.1.1.

The data policies to direct service-side traffic to the NAT are configured on two vSmart controllers, vSmart9 and vSmart10.

By default, OMP advertises all inside NAT pool IP addresses and all static NAT pool IP addresses, so all devices on the overlay network learn these routes automatically. In this example configuration, we configure OSPF and BGP to redistribute outside NAT pool IP addresses. The result is that OSPF on vEdge5 redistributes outside NAT pool IP addresses to its OSPF neighbor, Router7, and BGP redistributes outside NAT pool IP addresses to its BGP neighbor, Router8.

Configure Service-Side NAT on the vEdge Routers

vEdge5 and vEdge6 are vEdge routers at two different sites. They are both connected to the Internet, and they are both are running NAT.

On vEdge5, we configure a NAT pool that can translate four static addresses:

```
vEdge5(config)# vpn 1
vEdge5(config-vpn-1)# interface natpool1
vEdge5(config-natpool1)# ip address 10.15.1.4/30
vEdge5(config-natpool1)# no shutdown
```

With this configuration, the following IP addresses are available for static source IP address mapping: 10.15.1.4, 10.15.1.5, 10.15.1.6, and 10.15.1.7.

We then configure NAT on this interface:

```
vEdge5(config-natpool1)# nat
```

We want to enforce 1:1 static source IP address mapping:

```
vEdge5(config-nat)# no overload
```

If you omit this command, the default behavior is **overload** , which is effectively dynamic NAT. With the default behavior, all IP addresses are translated to an address in the pool of NAT addresses configured in the **ip address** command. The addresses are mapped one to one until the address pool is depleted. Then, the last address is used multiple times, and the port number is changed to a random value between 1024 and 65535. Overloading effectively implements dynamic NAT.

For this NAT pool, we want network address translation to be performed only on inside IP source addresses. Inside address translation is the default behavior. You can also explicitly configure it:

```
vEdge5(config-nat)# direction inside
```

For this example, we configure two NAT mappings. We want to NAT the source IP address 10.20.24.17, which is the IP address of Router7, This address is an inside address; that is, it is an address at the local site. We also want to NAT the source IP address 10.20.25.18, which comes from Router 8, a router behind vEdge6. This is an outside address.

```
vEdge5(config-nat)# static source-ip 10.20.24.17 translate-ip 10.15.1.4 inside
vEdge5(config-nat)# static source-ip 10.20.25.18 translate-ip 10.25.1.1 outside
```

We translate the inside source IP address 10.20.24.17 to 10.15.1.4. Because this NAT pool performs NAT only on inside IP source addresses (**direction inside**), and because 10.20.24.17 is an inside address, the translated address must be one of the addresses in the IP address range 10.15.1.4/30, which is the IP address of the NAT pool interface (configured in the **ip address** command).

We translate the outside address 10.20.25.18 to 10.25.1.1. Because this NAT pool performs NAT only on inside IP source addresses, we can translate outside addresses to any IP address that is routable on the service-side network behind vEdge5.

At vEdge6, we want to translate the source IP address 10.15.1.4, the translated address received from vEdge5, to an address that is routable on the service network behind vEdge6. The NAT pool that we configure on vEdge6 performs NAT only on outside addresses:

```
vEdge6(config)# vpn 1
vEdge6(config-vpn-1)# interface natpool2
vEdge6(config-natpool2)# ip address 10.16.1.4/30
vEdge6(config-natpool2)# no shutdown
vEdge6(config-natpool2)# nat
vEdge6(config-nat)# direction outside
vEdge6(config-nat)# static source-ip 10.15.1.4 translate-ip 10.16.1.4 outside
vEdge6(config-nat)# no overload
```

Here are the complete configurations for the static NAT pools on the vEdge5 and vEdge6 routers:

```
vEdge5# show running-config vpn 1 interface natpool1
vpn 1
 interface natpool1
  ip address 10.15.1.4/30
  nat
   static source-ip 10.20.24.17 translate-ip 10.15.1.4 inside
   static source-ip 10.20.25.18 translate-ip 10.25.1.1 outside
   no overload
  !
  no shutdown
 !
!
vEdge6# show running-config vpn 1 interface natpool2
vpn 1
 interface natpool2
  ip address 10.16.1.4/30
  nat
   static source-ip 10.15.1.4 translate-ip 10.16.1.4 outside
   direction outside
   no overload
  !
  no shutdown
```

```
 !
!
```

Configure Data Policies on vSmart Controllers

To direct service-side traffic to the NAT pool interface, you configure centralized data policies on the vSmart controllers. Our example network has two vSmart controllers, vSmart9 and vSmart10. The data policies must be identical on both of them.

The basic structure of the data policy is to define the match criteria for the packets destined to the NAT interface and then, in the action portion of the policy, to assign or direct the packets to a specific NAT pool. The data policy structure looks like this:

```
For a data-policy
  For a vpn-list
    For a sequence number
      Match specific criteria
      Action accept
        nat pool number
Apply the data-policy to all data traffic
```

In our example, we want a data policy that directs service-side traffic behind the vEdge5 router to the router's NAT pool interface 1 (**interface natpool 1**). Here is one portion of the data policy (specifically, one of the sequences within the policy) that does this, defining the service-side traffic by its source and destination IP addresses:

```
policy
  data-policy accept_nat
    vpn-list vpn_1
      sequence 108
        match
          source-ip      10.1.17.0/24
          destination-ip 10.25.1.0/24
        !
        action accept
          count nat_108
          nat pool 1
        !
      !
    !
  !
!
apply-policy
  site-list vedge1
    data-policy accept_nat all
  !
!
```

This data policy snippet takes data traffic whose source IP address is in the range 10.1.17.0/24 and destination is 10.25.1.0/24, accepts it, counts it to the file nat_108, and directs it to NAT pool 1. The source IP prefix 10.1.17.0/24 corresponds to traffic originating from Router7. The destination IP prefix 10.25.1.0/24 is an IP address that the vEdge5 router has translated from 10.20.25.18. This latter IP address corresponds to data traffic originating on Router8, so data traffic with a destination IP prefix of 10.25.1.0/24 is delivered to vEdge6 and then to Router8.

The second configuration snippet applies the policy to all traffic passing through the router (**all**).

The complete data policy configuration is shown at the end of this article.

Configure Route Redistribution by OSPF and BGP

By default, OSPF redistributes routes learned from inside NAT pool prefixes into OSPF. In addition, no routes from any other protocols are distributed into OSPF. The same is true for BGP.

In our configuration, we also want OSPF and BG to redistribute routes learned from outside NAT pool addresses. We also want OSPF to redistribute connected, OMP, and static routes, and we want BGP to redistribute OMP and static routes.

```
vEdge5# show running-config vpn 1 router
vpn 1
 router
  ospf
   redistribute static
   redistribute connected
   redistribute omp
   redistribute natpool-outside
   area 0
    interface ge0/4
     hello-interval 1
     dead-interval  3
    exit
   exit
  !
 !
!
vEdge6# show running-config vpn 1 router
vpn 1
 router
  bgp 1
   timers
    keepalive 1
    holdtime  3
   !
   address-family ipv4-unicast
    redistribute static
    redistribute omp
    redistribute natpool-outside
   !
   neighbor 10.20.25.18
    no shutdown
    remote-as 2
    timers
     connect-retry          2
     advertisement-interval 1
    !
   !
  !
 !
!
```

Verify the NAT Configuration

You use two commands to verify that the NAT configuration is operational: **show interface** and **show ip nat interface** .

The **show interface** command output indicates which NAT pool interfaces are configured and provides status about them. The command output for the vEdge5 router shows that NAT pool interface 1 is administratively and operational up. This command output also shows information about the other interfaces configured on vEdge 5.

```
vEdge5# show interface
                                       IF      IF
TCP
                AF                     ADMIN   OPER    ENCAP                                  SPEED
MSS                RX       TX
VPN  INTERFACE  TYPE  IP ADDRESS       STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR            MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------
-----------------------------------------------
0    ge0/0      ipv4  10.1.15.15/24     Up      Up      null   transport  1500  00:0c:29:7d:1e:fe  1000
```

```
full    1420    0:09:52:43  226385   228332
0    ge0/1      ipv4  10.1.17.15/24     Up      Up      null   service    1500  00:0c:29:7d:1e:08  1000
full    1420    0:07:00:23  1262     10
0    ge0/2      ipv4  -                 Down    Down    null   service    1500  00:0c:29:7d:1e:12  -
-       1420    -           0        0
0    ge0/3      ipv4  10.0.20.15/24     Up      Up      null   service    1500  00:0c:29:7d:1e:1c  1000
full    1420    0:07:00:23  1272     10
0    ge0/6      ipv4  57.0.1.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:3a  1000
full    1420    0:07:00:22  1262     9
0    ge0/7      ipv4  10.0.100.15/24    Up      Up      null   service    1500  00:0c:29:7d:1e:44  1000
full    1420    0:09:52:04  2931     741
0    system     ipv4  172.16.255.15/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10
full    1420    0:06:59:24  0        0
1    ge0/4      ipv4  10.20.24.15/24    Up      Up      null   service    1500  00:0c:29:7d:1e:26  1000
full    1420    0:07:00:19  26310    25065
1    ge0/5      ipv4  56.0.1.15/24      Up      Up      null   service    1500  00:0c:29:7d:1e:30  1000
full    1420    0:07:00:19  1261     8
1    natpool1   ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool7   ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool8   ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool9   ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool10  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool11  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool12  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool13  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool14  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool15  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
1    natpool16  ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:52:41  0        0
512  eth0       ipv4  10.0.1.15/24      Up      Up      null   service    1500  00:50:56:00:01:0f  1000
full    0       0:09:52:42  19482    12745
```

Similarly, on the vEdge6 router, we can check that its NAT pool 2 interface is up:

```
vEdge6# show interface
                                       IF      IF
TCP
                AF                     ADMIN   OPER    ENCAP                                  SPEED
MSS                 RX       TX
VPN  INTERFACE  TYPE  IP ADDRESS       STATUS  STATUS  TYPE   PORT TYPE  MTU   HWADDR            MBPS
DUPLEX  ADJUST  UPTIME      PACKETS  PACKETS
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------
0    ge0/0      ipv4  10.1.16.16/24     Up      Up      null   transport  1500  00:0c:29:d7:63:18  1000
full    1420    0:09:58:47  271786   294577
0    ge0/1      ipv4  10.1.18.16/24     Up      Up      null   service    1500  00:0c:29:d7:63:22  1000
full    1420    0:07:06:18  1274     10
0    ge0/2      ipv4  -                 Down    Down    null   service    1500  00:0c:29:d7:63:2c  -
-       1420    -           0        0
0    ge0/3      ipv4  10.0.21.16/24     Up      Up      null   service    1500  00:0c:29:d7:63:36  1000
full    1420    0:07:06:18  1285     9
0    ge0/7      ipv4  10.0.100.16/24    Up      Up      null   service    1500  00:0c:29:d7:63:5e  1000
full    1420    0:09:58:04  2971     746
0    system     ipv4  172.16.255.16/32  Up      Up      null   loopback   1500  00:00:00:00:00:00  10
```

```
full    1420    0:07:05:28  0        0
1    ge0/4      ipv4  10.20.25.16/24    Up      Up      null   service    1500  00:0c:29:d7:63:40  1000
full    1420    0:07:06:15  51457    50250
1    ge0/5      ipv4  60.0.1.16/24      Up      Up      null   service    1500  00:0c:29:d7:63:4a  1000
full    1420    0:07:06:15  1273     8
1    ge0/6      ipv4  61.0.1.16/24      Up      Up      null   service    1500  00:0c:29:d7:63:54  1000
full    1420    0:07:06:15  1255     8
1    natpool2   ipv4  -                 Up      Up      null   service    1500  00:00:00:00:00:00  10
full    1420    0:05:56:37  0        0
1    natpool12  ipv4  -                 Down    Down    null   service    1500  00:00:00:00:00:00  -
-       1420    -           0        0
512  eth0       ipv4  10.0.1.16/24      Up      Up      null   service    1500  00:50:56:00:01:10  1000
full    0       0:09:58:39  17650    11555
```

To display information about the NAT pools themselves, use the **show ip nat interface** command. Here is the command output for the vEdge5 router in tabular format and for vEdge6 in nontabular format:

```
vEdge5# show ip nat interface
                                                                          FIB                      NUMBER
                                                                  FILTER  FILTER                   IP
VPN  IFNAME     MAP TYPE              FILTER TYPE                 COUNT   COUNT   IP               POOLS
-------------------------------------------------------------------------------------------------------
1    natpool1   endpoint-independent  address-port-restricted     0       0       10.15.1.4/30     4
1    natpool7   endpoint-independent  address-port-restricted     0       0       10.21.26.15/32   1
1    natpool8   endpoint-independent  address-port-restricted     0       0       10.21.27.15/32   1
1    natpool9   endpoint-independent  address-port-restricted     0       0       10.21.28.15/32   1
1    natpool10  endpoint-independent  address-port-restricted     0       0       10.21.29.15/32   1
1    natpool11  endpoint-independent  address-port-restricted     0       0       10.21.30.15/32   1
1    natpool12  endpoint-independent  address-port-restricted     0       0       10.21.31.15/32   1
1    natpool13  endpoint-independent  address-port-restricted     0       0       10.21.32.15/32   1
1    natpool14  endpoint-independent  address-port-restricted     0       0       10.21.33.15/32   1
1    natpool15  endpoint-independent  address-port-restricted     0       0       10.21.34.15/32   1
1    natpool16  endpoint-independent  address-port-restricted     0       0       10.21.35.15/32   1
vEdge6# show ip nat interface
ip nat interface nat-vpn 1 nat-ifname natpool2
 mapping-type     endpoint-independent
 filter-type      address-port-restricted
 filter-count     0
 fib-filter-count 0
 ip               10.16.1.4/30
```

Verify Routes and Route Redistribution

We configured OSPF and BGP to redistributes routes learned from outside NAT into OSPF and BGP, respectively. (We also configured OSPF and BGP ro redistribute static and OMP routes, and we configured OMP to redistribute routes learned from directly connected devices.)

To see where routes have been learned from, look at the Protocol field in the output of the **show ip routes** command.

Looking on the vEdge5 router, we see that OSPF has redistributed 10.15.1.4/30, a route learned from an inside NAT (these routes are redistributed by default) and 10.25.1.1/32, a route learned from an outside NAT. The vEdge5 router translates the IP address 10.25.1.1 from 10.20.25.18. Both these routes have a next-hop interface of natpool1, which is the NAT pool we configured to run static NAT.

```
vEdge5# show ip routes
Codes Proto-sub-type:
  IA -> ospf-inter-area,
  E1 -> ospf-external1, E2 -> ospf-external2,
  N1 -> ospf-nssa-external1, N2 -> ospf-nssa-external2,
  e -> bgp-external, i -> bgp-internal
Codes Status flags:
  F -> fib, S -> selected, I -> inactive,
  B -> blackhole, R -> recursive

                                          PROTOCOL  NEXTHOP     NEXTHOP          NEXTHOP
```

```
VPN    PREFIX              PROTOCOL         SUB TYPE  IF NAME     ADDR            VPN      TLOC IP
COLOR            ENCAP  STATUS
-------------------------------------------------------------------------------------------------------------
-----------------------------------
0      0.0.0.0/0           static           -         ge0/0       10.1.15.13      -        -
-                -      F,S
0      10.0.20.0/24        connected        -         ge0/3       -               -        -
-                -      F,S
0      10.0.100.0/24       connected        -         ge0/7       -               -        -
-                -      F,S
0      10.1.15.0/24        connected        -         ge0/0       -               -        -
-                -      F,S
0      10.1.17.0/24        connected        -         ge0/1       -               -        -
-                -      F,S
0      57.0.1.0/24         connected        -         ge0/6       -               -        -
-                -      F,S
0      172.16.255.15/32    connected        -         system      -               -        -
-                -      F,S
1      1.1.1.0/24          omp              -         -           -               -        172.16.255.14
lte              ipsec  F,S
1      2.2.0.0/16          static           -         -           -               -        -
-                -      B,F,S
1      4.4.4.4/32          static           -         -           -               -        -
-                -      B,F,S
1      9.0.0.0/8           omp              -         -           -               -        172.16.255.16
lte              ipsec  F,S
1      10.1.17.0/24        static           -         ge0/4       10.20.24.17     -        -
-                -      F,S
1      10.1.18.0/24        omp              -         -           -               -        172.16.255.16
lte              ipsec  F,S
1      10.2.2.0/24         omp              -         -           -               -        172.16.255.11
lte              ipsec  F,S
1      10.2.3.0/24         omp              -         -           -               -        172.16.255.21
lte              ipsec  F,S
1      10.15.1.4/30        natpool-inside   -         natpool1    -               -        -
-                -      F,S
1      10.20.24.0/24       ospf             -         ge0/4       -               -        -
-                -      -
1      10.20.24.0/24       connected        -         ge0/4       -               -        -
-                -      F,S
1      10.20.25.0/24       omp              -         -           -               -        172.16.255.16
lte              ipsec  F,S
1      10.25.1.0/24        static           -         -           -               -        -
-                -      B,F,S
1      10.25.1.1/32        natpool-outside  -         natpool1    -               -        -
-                -      F,S
1      56.0.1.0/24         connected        -         ge0/5       -               -        -
-                -      F,S
1      60.0.1.0/24         omp              -         -           -               -        172.16.255.16
lte              ipsec  F,S
1      61.0.1.0/24         omp              -         -           -               -        172.16.255.16
lte              ipsec  F,S
512    10.0.1.0/24         connected        -         eth0        -               -        -
-                -      F,S
```

The vEdge6 router translates the outside source IP address 10.15.1.4 to 10.16.1.4. The route table on vEdge6 shows this route and that it has been learned from an outside NAT. The next-hop interface for this prefix is natpool2.

```
vEdge6# show ip routes
Codes Proto-sub-type:
  IA -> ospf-inter-area,
  E1 -> ospf-external1, E2 -> ospf-external2,
  N1 -> ospf-nssa-external1, N2 -> ospf-nssa-external2,
  e -> bgp-external, i -> bgp-internal
```

```
Codes Status flags:
  F -> fib, S -> selected, I -> inactive,
  B -> blackhole, R -> recursive

                                                PROTOCOL  NEXTHOP     NEXTHOP          NEXTHOP
VPN    PREFIX              PROTOCOL             SUB TYPE  IF NAME     ADDR             VPN      TLOC IP
COLOR            ENCAP  STATUS
-------------------------------------------------------------------------------------------------------------
-----------------------------------
0      0.0.0.0/0           static               -         ge0/0       10.1.16.13       -        -
-                -      F,S
0      10.0.21.0/24        connected            -         ge0/3       -                -        -
-                -      F,S
0      10.0.100.0/24       connected            -         ge0/7       -                -        -
-                -      F,S
0      10.1.16.0/24        connected            -         ge0/0       -                -        -
-                -      F,S
0      10.1.18.0/24        connected            -         ge0/1       -                -        -
-                -      F,S
0      172.16.255.16/32    connected            -         system      -                -        -
-                -      F,S
1      1.1.1.0/24          omp                  -         -           -                -        172.16.255.14
lte              ipsec  F,S
1      2.2.0.0/16          static               -         -           -                -        -
-                -      B,F,S
1      4.4.4.4/32          omp                  -         -           -                -        172.16.255.15
lte              ipsec  F,S
1      9.0.0.0/8           static               -         -           -                -        -
-                -      B,F,S
1      10.1.17.0/24        omp                  -         -           -                -        172.16.255.15
lte              ipsec  F,S
1      10.1.18.0/24        static               -         ge0/4       10.20.25.18      -        -
-                -      F,S
1      10.2.2.0/24         omp                  -         -           -                -        172.16.255.11
lte              ipsec  F,S
1      10.2.3.0/24         omp                  -         -           -                -        172.16.255.21
lte              ipsec  F,S
1      10.15.1.4/30        omp                  -         -           -                -        172.16.255.15
lte              ipsec  F,S
1      10.16.1.4/30        natpool-outside      -         natpool2    -                -        -
-                -      F,S
1      10.20.24.0/24       omp                  -         -           -                -        172.16.255.15
lte              ipsec  F,S
1      10.20.25.0/24       connected            -         ge0/4       -                -        -
-                -      F,S
1      10.25.1.0/24        omp                  -         -           -                -        172.16.255.15
lte              ipsec  F,S
1      56.0.1.0/24         omp                  -         -           -                -        172.16.255.15
lte              ipsec  F,S
1      60.0.1.0/24         connected            -         ge0/5       -                -        -
-                -      F,S
1      61.0.1.0/24         connected            -         ge0/6       -                -        -
-                -      F,S
512    10.0.1.0/24         connected            -         eth0        -                -        -
-                -      F,S
```

View Interface Statistics

To display packet receipt and transmission statistics for the interfaces, use the **show interface statistics** command. The output shows the following statistics:

```
vEdge5# show interface statistics natpool1 | notab
interface vpn 1 interface natpool1 af-type ipv4
```

```
 rx-packets 0
 rx-octets  0
 rx-errors  0
 rx-drops   0
 tx-packets 0
 tx-octets  0
 tx-errors  0
 tx-drops   0
 rx-pps     0
 rx-kbps    0
 tx-pps     0
 tx-kbps    0
```

To display NAT-specific interface statistics, use the **show ip nat interface-statistics** command. The output shows the following statistics for each NAT pool:

```
vEdge5# show ip nat interface-statistics
ip nat interface-statistics nat-vpn 1 nat-ifname natpool1
 nat-outbound-packets      0
 nat-inbound-packets       0
 nat-encode-fail           0
 nat-decode-fail           0
 nat-map-add-fail          0
 nat-filter-add-fail       0
 nat-filter-lookup-fail    0
 nat-state-check-fail      0
 nat-policer-drops         0
 outbound-icmp-error       0
 inbound-icmp-error        0
 inbound-icmp-error-drops  0
 nat-fragments             0
 nat-fragments-fail        0
 nat-unsupported-proto     0
 nat-map-no-ports          0
 nat-map-cannot-xlate      0
 nat-filter-map-mismatch   0
 nat-map-ip-pool-exhausted 0
```

View the Data Policy Pushed to the vEdge Routers

To view and verify the data policy pushed from the vSmart controllers to the two vEdge routers, use the **show policy from-vsmart** command. The following is the command output for the vEdge5 router. The output on vEdge6 is identical.

```
vEdge5# show policy from-vsmart
from-vsmart data-policy accept_nat
direction all
vpn-list vpn_1
  sequence 100
   match
    source-ip      10.20.24.0/24
    destination-ip 10.20.25.0/24
   action accept
    count nat
    nat pool 1
  sequence 101
   match
    source-ip      10.20.24.0/24
    destination-ip 10.1.15.13/32
   action accept
    count nat_inet
    nat use-vpn 0
  sequence 102
```

```
     match
      dscp 15
     action accept
      count nat_dscp
      nat use-vpn 0
    sequence 104
     match
      source-ip      10.1.18.0/24
      destination-ip 10.20.24.0/24
     action accept
      count nat2
      nat pool 1
    sequence 105
     match
      source-ip      10.1.18.0/24
      destination-ip 10.1.17.0/24
     action accept
      count nat3
      nat pool 1
    sequence 106
     match
      source-ip      10.1.17.0/24
      destination-ip 10.20.25.0/24
     action accept
      nat pool 1
    sequence 107
     match
      source-ip      10.15.1.0/24
      destination-ip 10.20.25.0/24
     action accept
      nat pool 2
    sequence 108
     match
      source-ip      10.1.17.0/24
      destination-ip 10.25.1.0/24
     action accept
      count nat_108
      nat pool 1
    sequence 109
     match
      source-ip      10.20.24.0/24
      destination-ip 10.25.1.0/24
     action accept
      count nat_109
      nat pool 1
   default-action accept
from-vsmart lists vpn-list vpn_1
vpn 1
```

Configurations for Each Network Device

For each of the network devices in this configuration example, this section shows the portions of the configuration relevant to the service-side NAT configuration.

vEdge5 Router

The vEdge5 router is located at site 500, has a system IP address of 172.16.255.15, and has one connection to the Internet:

```
system
 host-name         vm5
```

```
 system-ip         172.16.255.15
 site-id           500
!
vpn 0
 interface ge0/0
  ip address 10.1.15.15/24
  tunnel-interface
   encapsulation ipsec
   color lte
   hello-interval  60000
   hello-tolerance 120
   no allow-service bgp
   allow-service dhcp
   allow-service dhcpv6
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
  !
  no shutdown
 !
!
```

In VPN 1, NAT pool 1 runs 1:1 static NAT:

```
vpn 1
 interface natpool1
  ip address 10.15.1.4/30
  nat
   static source-ip 10.20.24.17 translate-ip 10.15.1.4 inside
   static source-ip 10.20.25.18 translate-ip 10.25.1.1 outside
   no overload
  !
  no shutdown
 !
```

VPN 1 also has a number of other NAT pool interfaces:

```
 interface natpool10
  ip address 10.21.29.15/32
  no shutdown
 !
 interface natpool11
  ip address 10.21.30.15/32
  no shutdown
 !
 interface natpool12
  ip address 10.21.31.15/32
  no shutdown
 !
 interface natpool13
  ip address 10.21.32.15/32
  no shutdown
 !
 interface natpool14
  ip address 10.21.33.15/32
  no shutdown
 !
 interface natpool15
  ip address 10.21.34.15/32
  no shutdown
 !
```

```
 interface natpool16
  ip address 10.21.35.15/32
  no shutdown
 !
 interface natpool7
  ip address 10.21.26.15/32
  no shutdown
 !
 interface natpool8
  ip address 10.21.27.15/32
  no shutdown
 !
 interface natpool9
  ip address 10.21.28.15/32
  no shutdown
 !
 ip route 2.2.0.0/16 null0
 ip route 4.4.4.4/32 null0
 ip route 10.1.17.0/24 10.20.24.17
 ip route 10.25.1.0/24 null0
!
```

OSPF runs in VPN 1 and is configured to redistribute routes learned from outside NAT prefixes into OSPF:

```
vpn 1
 router
  ospf
   timers spf 200 1000 10000
   redistribute static
   redistribute connected
   redistribute omp
   redistribute natpool-outside
   area 0
    interface ge0/4
     hello-interval 1
     dead-interval  3
    exit
   exit
  !
 !
!
```

vEdge6 Router

The vEdge6 router is located at site 600, has a system IP address of 172.16.255.16, and has one connection to the Internet:

```
system
 host-name         vm6
 system-ip         172.16.255.16
 site-id           600
!
vpn 0
 interface ge0/0
  ip address 10.1.16.16/24
  tunnel-interface
   encapsulation ipsec
   color lte
   no allow-service bgp
   allow-service dhcp
   allow-service dhcpv6
   allow-service dns
   allow-service icmp
   no allow-service sshd
```

```
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
  !
  no shutdown
 !
!
```

VPN 1 has one NAT pool for static address translation:

```
vpn 1
 interface natpool12
  ip address 10.1.155.4/30
  shutdown
 !
 interface natpool2
  ip address 10.16.1.4/30
  nat
   static source-ip 10.15.1.4 translate-ip 10.16.1.4 outside
   direction outside
   no overload
  !
  no shutdown
 !
 ip route 2.2.0.0/16 null0
 ip route 9.0.0.0/8 null0
 ip route 10.1.18.0/24 10.20.25.18
!
```

BGP runs in VPN 1 and is configured to redistribute routes learned from outside NAT prefixes into BGP:

```
vm6# show running-config vpn 1 router
vpn 1
 router
  bgp 1
   timers
    keepalive 1
    holdtime  3
   !
   address-family ipv4-unicast
    redistribute static
    redistribute omp
    redistribute natpool-outside
   !
   neighbor 10.20.25.18
    no shutdown
    remote-as 2
    timers
     connect-retry          2
     advertisement-interval 1
    !
   !
  !
 !
!
```

Router7 and Router8

Router7 sits in the local site behind the vEdge5 router, and it is an OSPF peer with vEdge5. Router8 sits behind the vEdge6 router and is an IBGP peer with vEdge6.

In our example network, both these routers are configured on vEdge software routers. However, there is nothing in their configuration that specifically relates to static NAT, so we do not show the configurations for these two devices.

vSmart9 and vSmart10 vSmart Controllers

You configure the data policy that runs on the vEdge routers to direct data traffic to the NAT interfaces on the vSmart controllers. The vSmart controllers then push the data policy to the appropriate vEdge routers. The configure data policy must be identical on all vSmart controllers in the overlay network to ensure reproducible data traffic handling in the network.

Here is the complete policy configuration for the two vSmart controllers in our example:

```
policy
 data-policy accept_nat
  vpn-list vpn_1
   sequence 100
    match
     source-ip      10.20.24.0/24
     destination-ip 10.20.25.0/24
    !
    action accept
     count nat
     nat pool 1
    !
   !
   sequence 101
    match
     source-ip      10.20.24.0/24
     destination-ip 10.1.15.13/32
    !
    action accept
     count nat_inet
     nat use-vpn 0
    !
   !
   sequence 102
    match
     dscp 15
    !
    action accept
     count nat_dscp
     nat use-vpn 0
    !
   !
   sequence 104
    match
     source-ip      10.1.18.0/24
     destination-ip 10.20.24.0/24
    !
    action accept
     count nat2
     nat pool 1
    !
   !
   sequence 105
    match
     source-ip      10.1.18.0/24
     destination-ip 10.1.17.0/24
    !
    action accept
     count nat3
     nat pool 1
    !
```

```
    !
    sequence 106
     match
      source-ip      10.1.17.0/24
      destination-ip 10.20.25.0/24
     !
     action accept
      nat pool 1
     !
    !
    sequence 107
     match
      source-ip      10.15.1.0/24
      destination-ip 10.20.25.0/24
     !
     action accept
      nat pool 2
     !
    !
    sequence 108
     match
      source-ip      10.1.17.0/24
      destination-ip 10.25.1.0/24
     !
     action accept
      count nat_108
      nat pool 1
     !
    !
       sequence 109
     match
      source-ip      10.20.24.0/24
      destination-ip 10.25.1.0/24
     !
     action accept
      count nat_109
      nat pool 1
     !
    !
    default-action accept
   !
  !
  lists
   vpn-list vpn_1
    vpn 1
   !
   site-list east
    site-id 100
    site-id 500
   !
   site-list vedge1
    site-id 500
   !
   site-list vedge2
    site-id 600
   !
   site-list vedges
    site-id 500
    site-id 600
   !
   site-list west
    site-id 200
    site-id 400
    site-id 600
```

```
     !
     prefix-list prefix_list
      ip-prefix 10.20.24.0/24
     !
    !
  !
  vm9# show running-config apply-policy
  apply-policy
   site-list vedge1
    data-policy accept_nat all
   !
   site-list vedge2
    data-policy accept_nat all
   !
  !
```

Additional Information

Configuring Service-Side NAT
Using a vEdge Router as a NAT Device

Using Umbrella DNS Security

For DNS-layer security, you can configure a vEdge router to act as a DNS forwarder to Cisco Umbrella. Umbrella is a cloud-delivered service for provisioning secure and compliant guest Wi-Fi.

You configure Umbrella DNS security with a centralized data policy. You create the policy on a vSmart controller, and it is pushed to the vEdge routers. This policy ensures that the router enables DNS-level security for all end points, both controlled and uncontrolled, in your branch environment through Cisco Umbrella by transparently intercepting DNS queries and forwarding them to Umbrella DNS.

In order to apply the DNS redirect policies to Umbrella, you must first register the public IP addresses of the remote sites on Umbrella portal, under the Identities ► Network option.

CLI Configuration Procedure

The following high-level steps show the minimum policy components required to redirect DNS traffic to Cisco Umbrella:

1. Create one or more lists of overlay network sites to which the centralized data policy is to be applied (in an **apply-policy** command):
   vSmart(config)# **policy**
   vSmart(config-policy)# **lists site-list** *list-name*
   vSmart(config-lists)# **site-id** *site-id*
   The list can contain as many site IDs as necessary. Include one **site-id** command for each site ID. For contiguous site IDs, you can specify a range of numbers separated with a dash (–).

2. Create lists of VPNs to which to apply the Umbrella DNS redirect policy (in a **policy data-policy** command):
   vSmart(config)# **policy lists**
   vSmart(config-lists)# **vpn-list** *list-name*
   vSmart(config-lists)# **vpn** *vpn-id*

3. Create a data policy instance and associate it with a list of VPNs:
   vSmart(config)# **policy data-policy** *policy-name*
   vSmart (config-data-policy)# **vpn-list** *list-name*

4. Create one match–action pair sequence:
   vSmart(config-vpn-list)# **sequence** *number1*
   The match–action pairs are evaluated in order, by sequence number, starting with the lowest numbered pair and ending when the route matches the conditions in one of the pairs. Or if no match occurs, the default action is taken (either rejecting the route or accepting it as is).

5. Configure a match condition to process DNS requests to be forwarded to Umbrella:
   vSmart(config-sequence1)# **match dns request**

6. Configure the DNS redirect action, specifying the IP address of the Umbrella DNS service, either 208.67.222.222 or 208.67.220.220:
   vSmart(config-sequence1)# **action accept redirect-dns** *ip-address*

7. Create a second match–action pair sequence:
   vSmart(config-vpn-list)# **sequence** *number2*

8. Configure a match condition to process DNS responses from the Umbrella service:
   vSmart(config-sequence2)# **match dns response**

9. Configure the redirect DNS host so that the DNS response can be correctly forwarded back to the service VPN:
   vSmart(config-sequence2)# **action accept redirect-dns host**

10. Apply the policy to a list of sites in the overlay network:
    vSmart(config)# **apply-policy site-list** *list-name* **data-policy** *policy-name* (**from-service** | **from-tunnel**)

Structural Components of Policy Configuration for Umbrella DNS

Below are the structural components required to configure DNS traffic redirection to Cisco Umbrella. You configure this policy on a vSmart controller. The components related to configuring umbrella DNS are explained in the sections below. For an explanation of the data policy components that are not specifically related to DNS traffic redirection, see Configuring Centralized Data Policy .

```
policy
  lists
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn-id vpn-id
 data-policy policy-name
   vpn-list list-name
     sequence number
       match
         dns (request | response)
       action accept
         count counter-name
         log
         redirect-dns (ip-address | host)
     default-action
       (accept | drop)
apply-policy
  site-list list-name data-policy policy-name (from-service | from-tunnel)
```

Lists

A data policy for configuring a DNS forwarder to Cisco Umbrella uses the following types of lists to group related items. You configure these lists under the **policy lists** command hierarchy on vSmart controllers.

List Type	Description	Command
Sites	List of one or more site identifiers in the overlay network. To configure multiple sites in a single list, include multiple **site-id** options, specifying one site number in each option.You can specify a single site identifier (such as **site-id 1**) or a range of site identifiers (such as **site-id 1-10**).	**site-list** *list-name* **site-id** *site-id*

VPNs	List of one or more VPNs in the overlay network. To configure multiple VPNs in a single list, include multiple **vpn** options, specifying one VPN number in each option. You can specify a single VPN identifier (such as **vpn-id 1**) or a range of VPN identifiers (such as **vpn-id 1-10**).	**vpn-list** *list-name* **vpn** *vpn-id*

In the vSmart controller configuration, you can create multiple iterations of each type of list. For example, it is common to create multiple site lists and multiple VPN lists so that you can apply data policy to different sites and different customer VPNs across the network.

When you create multiple iterations of a type of list (for example, when you create multiple VPN lists), you can include the same values or overlapping values in more than one of these list. You can do this either on purpose, to meet the design needs of your network, or you can do this accidentally, which might occur when you use ranges to specify values. (You can use ranges to specify data prefixes, site identifiers, and VPNs.) Here are two examples of lists that are configured with ranges and that contain overlapping values:

- **vpn-list list-1 vpn 1-10**
  vpn-list list-2 vpn 6-8
- **site-list list-1 site 1-10**
  site-list list-2 site 5-15

When you configure data policies that contain lists with overlapping values, or when you apply data policies, you must ensure that the lists included in the policies, or included when applying the policies, do not contain overlapping values. To do this, you must manually audit your configurations. The software performs no validation on the contents of lists, on the data policies themselves, or on how the policies are applied to ensure that there are no overlapping values.

If you configure or apply data policies that contain lists with overlapping values to the same site, one policy is applied and the others are ignored. Which policy is applied is a function of the internal behavior of Viptela software when it processes the configuration. This decision is not under user control, so the outcome is not predictable.

VPN Lists

Each data policy instance is associated with a VPN list. You configure VPN lists with the **policy data-policy vpn-list** command. The VPN list you specify must be one that you created with a **policy lists vpn-list** command.

Sequences

Within each VPN list, a data policy contains sequences of match–action pairs. The sequences are numbered to set the order in which data traffic is analyzed by the match–action pairs in the policy. You configure sequences with the **policy data-policy vpn-list sequence** command.

A sequence in a policy can contain one **match** command and one **action** command. Therefore, to configure a DNS forwarder to Cisco Umbrella, you must configure a pair of sequences, one for processing DNS requests to be forwarded to Cisco Umbrella and the second to process DNS responses from Cisco Umbrella.

Match Parameters

For a data policy for configuring a DNS forwarder to Cisco Umbrella, you must configure the following two match conditions. You configure the match parameters with the **match** command under the **policy data-policy vpn-list sequence** command hierarchy on vSmart controllers.

Description	Command	Value or Range
DNS requests to be forwarded to Cisco Umbrella	**dns request**	—
DNS responses from Cisco Umbrella	**dns response**	—

Action Parameters

When data traffic matches the match parameters, the specified action is applied to it. You configure the action parameters with the **action** command under the **policy data-policy vpn-list sequence** command hierarchy on vSmart controllers.

For a centralized data policy that configure a vEdge router to act as a DNS forwarder to Cisco Umbrella, configure the following actions. You can configure other actions, as described in Configuring Centralized Data Policy .

Description	Command	Value or Range
Redirect DNS requests to a Cisco Umbrella server. Specify this action for a **dns request** match condition.	**redirect-dns** *ip-address*	IP address of a Cisco Umbrella server, either 208.67.222.222 or 208.67.220.220.
Process DNS responses from a Cisco Umbrella server. Specify this action for a **dns response** match condition.	**redirect-dns host**	Forward the responses to the requesting service VPN.

Default Action

If a data packet being evaluated does not match any of the match conditions in a policy, a default action is applied. By default, the data packet is dropped. To modify this behavior, include the **policy data-policy vpn-list default-action accept** command.

Apply a Policy

For the centralized data policy to take effect so that the vEdge router can act as a DNS forwarder to Cisco Umbrella, you apply it to a list of sites in the overlay network. Because the policy configures the router to both forward packets to and receive packets from a Cisco Umbrella awecwe, specify the **all** option:

```
vSmart(config)# apply-policy site-list list-name data-policy policy-name (from-service | from-tunnel)
```

Example Configuration

The following example shows a data policy that enables Umbrella DNS security and that counts DNS traffic:

```
vSmart# show running-config policy
policy
 data-policy umbrella_dns
  vpn-list vpn_1
   sequence 1
    match
     dns          request
    !
    action accept
     count umbrella_traffic_outbound
     redirect-dns 208.67.220.220
    !
   !
  !
  sequence 2
    match
     dns          response
    !
    action accept
     count umbrella_traffic_inbound
     redirect-dns host
    !
   !
 lists
  vpn-list vpn_1
```

```
    vpn 1
   !
   site-list vedge1
    site-id 500
   !
  !
!
vSmart# show running-config apply-policy
apply-policy
  site-list vedge1 data-policy umbrella_dns from-service
```

Additional Information

Configuring Centralized Data Policy

Policy Applications CLI Reference

CLI commands for configuring and monitoring policy applications.

Application-Aware Routing Command Hierarchy

Configure and apply the policy on vSmart controllers:

```
policy
  lists
    app-list list-name
      (app application-name | app-family application-family)
    data-prefix-list list-name
      ip-prefix prefix/length
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn vpn-id
  sla-class sla-class-name
    jitter milliseconds
    latency milliseconds
    loss percentage
policy
  app-route-policy policy-name
    vpn-list list-name
      default-action sla-class sla-class-name
      sequence number
        match
          app-id app-id-name
          destination-data-prefix-list list-name
          destination-ip prefix/length
          destination-port number
          dns (request | response)
          dns-app-list list-name
          dscp number
          plp (high | low)
          protocol number
          source-data-prefix-list list-name
          source-ip prefix/length
          source-port address
        action
          backup-sla-preferred-color colors
          count
          log
          sla-class sla-class-name [strict] [preferred-color colors]
```

```
      default-action
        sla-class sla-class-name
apply-policy site-list list-name
  app-route-policy policy-name
```

Configure the data plane tunnel performance monitoring parameters on the vEdge router:

```
bfd
  app-route
    multiplier number
    poll-interval milliseconds
```

Cflowd Traffic Flow Monitoring Command Hierarchy

Configure on vSmart controllers only:

```
policy
  lists
    prefix-list list-name
      ip-prefix prefix/length
    site-list list-name
      site-id site-id
    vpn-list list-name
      vpn vpn-id
  cflowd-template template-name
    collector vpn vpn-id address ip-address port port-number transport transport-type
    flow-active-timeout seconds
    flow-inactive-timeout seconds
    flow-sampling-interval number
    template-refresh seconds
policy
  data-policy policy-name
    default-action action
    sequence number
      match
        destination-data-prefix-list list-name
        destination-ip prefix/length
        destination-port number
        dscp number
        protocol number
        source-data-prefix-list list-name
        source-ip prefix/length
        source-port address
      action
        count counter-name
        drop
        accept
          cflowd
apply-policy
  site-list list-name
    data-policy policy-name
    cflowd-template template-name
```

Local Internet Exit Command Hierarchy

Configure and apply a centralized data policy on the vSmart controller:

```
policy
  lists
    prefix-list list-name
      ip-prefix prefix/length
```

```
      site-list list-name
        site-id site-id
      vpn-list list-name
        vpn vpn-id
    cflowd-template template-name
      collector vpn vpn-id address ip-address port port-number
      flow-active-timeout seconds
      flow-inactive-timeout seconds
      template-refresh seconds
  policy
    data-policy policy-name
      default-action action
      sequence number
        match
          destination-data-prefix-list list-name
          destination-ip prefix/length
          destination-port number
          dscp number
          protocol number
          source-data-prefix-list list-name
          source-ip prefix/length
          source-port address
        action
          count counter-name
          drop
          accept
            nat use-vpn 0
  apply-policy
    site-list list-name
      data-policy policy-name
```

On a vEdge router, enable NAT functionality in the WAN VPN:

```
vpn vpn-id
  interface interface-name
    nat
      refresh (bi-directional | outbound)
      tcp-timeout minutes
      udp-timeout minutes
```

Operational Commands

clear app cflowd flow-all (on vEdge routers only)
clear app cflowd flows (on vEdge routers only)
clear app cflowd statistics (on vEdge routers only)
show app-route stats on vEdge routers only)
show app cflowd collector (on vEdge routers only)
show app cflowd flow-count (on vEdge routers only)
show app cflowd flows (on vEdge outers only)
show app cflowd statistics (on vEdge routers only)
show app cflowd template (on vEdge routers only)
show ip routes (on vEdge routers)
show policy from-vsmart (on vEdge routers only)
show running-config (on vSmart controllers only)

Additional Information

Application-Aware Routing
Policy Basics CLI Reference

Traffic Flow Monitoring with Cflowd
Using a vEdge Router as a NAT Device

Forwarding and QoS

Forwarding and QoS Overview

Forwarding is the transmitting of data packets from one vEdge router to another. Once the control plane connections of the Viptela overlay network are up and running, data traffic flows automatically over the IPsec connections between vEdge routers. Because data traffic never goes to or through the centralized vSmart controller, forwarding only occurs between the vEdge routers as they send and receive data traffic.

While the routing protocols running in the control plane provide a vEdge router the best route to reach the network that is on the service side of a remote vEdge router, there will be situations where it is beneficial to select more specific routes. Using forwarding, there are ways you can affect the flow of data traffic. Forwarding takes the data packet and sends it over the transport to the remote side, specifying what to do with the packet. It specifies the interface through which packets are sent to reach the service side of a remote vEdge router.

To modify the default data packet forwarding flow, you create and apply centralized data policy or localized data policy. With centralized data policy, you can manage the paths along which traffic is routed through the network, and you can permit or block traffic based on the address, port, and DSCP fields in the packet's IP header. With localized data policy, you can control the flow of data traffic into and out of a vEdge router's interfaces, enabling features such as quality of service (QoS) and mirroring. Note that QoS is synonymous with class of service (CoS).

Default Behavior without Data Policy

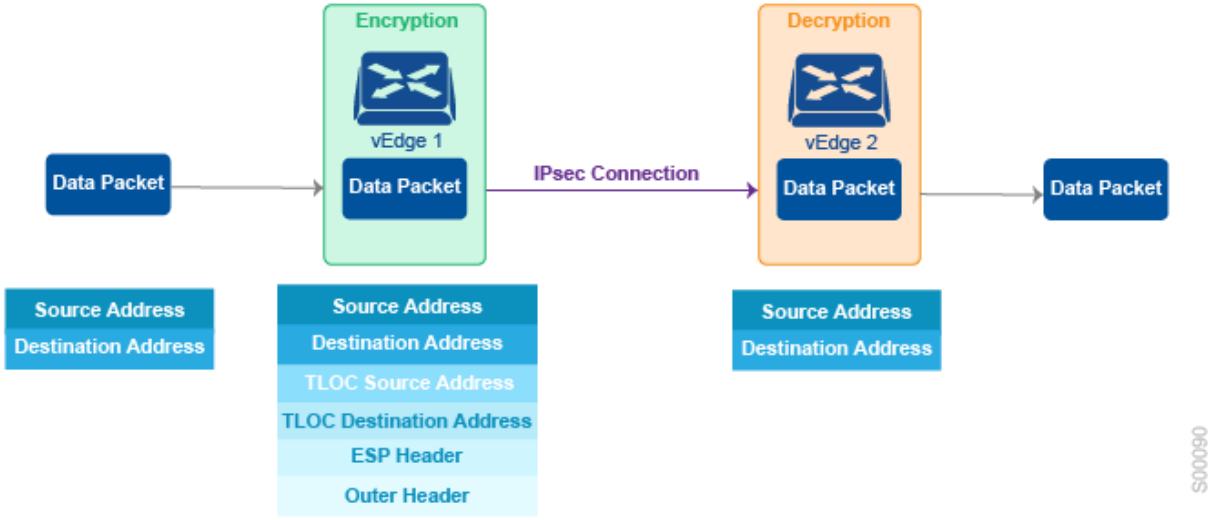
When no centralized data policy is configured on the vSmart controller, all data traffic is transmitted from the local service-side network to the local vEdge router, and then to the remote vEdge router and the remote service-side network, with no alterations in its path. When no access lists are configured on the local vEdge router to implement QoS or mirroring, the data traffic is transmitted to its destination with no alterations to its flow properties.

Let's follow the process that occurs when a data packet is transmitted from one site to another when no data policy of any type is configured:

- A data packet arriving from the local service-side network and destined for the remote service-side network comes to the vEdge-1 router. The packet has a source IP address and a destination IP address.
- The vEdge router looks up the outbound SA in its VPN route table, and the packet is encrypted with SA and gets the local TLOC. (The vEdge router previously received its SA from the vSmart controller. There is one SA per TLOC. More specifically, each TLOC has two SAs, an outbound SA for encryption and an inbound SA for decryption.)
- ESP adds an IPsec tunnel header to the packet.
- An outer header is added to the packet. At this point, the packet header has these contents: TLOC source address, TLOC destination address, ESP header, destination IP address, and source IP address.
- The vEdge router checks the local route table to determine which interface the packet should use to reach its destination.
- The data packet is sent out on the specified interface, onto the network, to its destination. At this point, the packet is being transported within an IPsec connection.
- When the packet is received by the vEdge router on the remote service-side network, the TLOC source address and TLOC destination address header fields are removed, and the inbound SA is used to decrypt the packet.

- The remote vEdge router looks up the destination IP address in its route table to determine the interface to use to reach to the service-side destination.

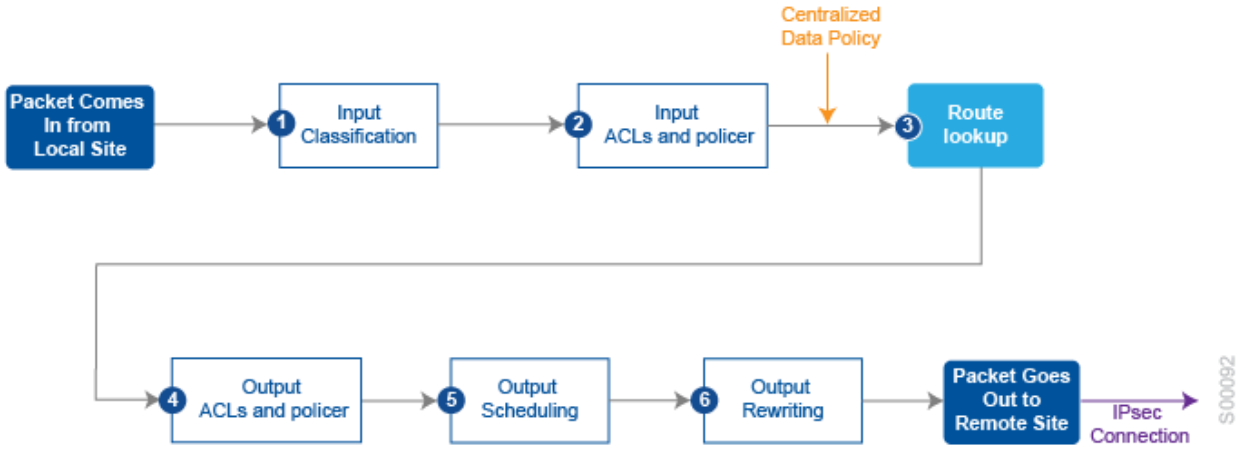
The figure below details this process.



Behavior Changes with QoS Data Policy

When you want to modify the default packet forwarding flow, you design and provision QoS policy. To activate the policy, you apply it to specific interfaces in the overlay network in either the inbound or the outbound direction. The direction is with respect to the vEdge routers in the network. You can have policies for packets coming in on an interface or for packets going out of an interface.

The figure below illustrates the QoS policies that you can apply to a data packet as it is transmitted from one branch to another. The policies marked Input are applied on the inbound interface to the vEdge router, and the policies marked Output are applied on the outbound interface to the vEdge router, before the packets are transmitted out the IPSec tunnel.



The table below describes each of the above steps.

Step	Description	Command

1	Define class map to classify packets, by importance, into appropriate forwarding classes. Reference the class map in an access list.	**class-map**
2	Define policer to specify the rate at which traffic is sent on the interface. Reference the policer in an access list. Apply the access list on an inbound interface.	**policer**
3	vEdge router checks the local route table to determine which interface the packet should use to reach its destination.	N/A
4	Define policer and reference the policer in an access list. Apply the access list on an outbound interface.	**policer**
5	Define QoS map to define the priority of data packets. Apply the QoS map on the outbound interface.	**qos-map**
6	Define rewrite-rule to overwrite the DSCP field of the outer IP header. Apply the rewrite-rule on the outbound interface.	**rewrite-rule**

Understanding How QoS Works

The QoS feature on the vEdge routers works by examining packets entering at the edge of the network. With localized data policy, also called access lists, you can provision QoS to classify incoming data packets into multiple forwarding classes based on importance, spread the classes across different interface queues, and schedule the transmission rate level for each queue. Access lists can be applied either in the outbound direction on the interface (as the data packet travels from the local service-side network into the IPsec tunnel toward the remote service-side network) or in the inbound direction (as data packets are exiting from the IPsec tunnel and being received by the local vEdge router.

To provision QoS, you must configure each vEdge router in the network. Generally, each router on the local service-side network examines the QoS settings of the packets that enter it, determines which packets are transmitted first, and processes the transmission based on those settings. As packets leave the network on the remote service-side network, you can rewrite the QoS bits of the packets before transmitting them to meet the policies of the targeted peer router.

The QoS behavior is as follows:

1. Default behavior is that TOS/DSCP is copied to outer header.
2. CS6 is used for the control packet.
3. If data packet has CS6, it rewrites to CS2.

Classify Data Packets

You can classify incoming traffic by associating each packet with a forwarding class. Forwarding classes group data packets for transmission to their destination. Based on the forwarding class, you assign packets to output queues. The vEdge routers service the output queues according to the associated forwarding, scheduling, and rewriting policies you configure.

Schedule Data Packets

You can configure a QoS map for each output queue to specify the bandwidth, delay buffer size, and packet loss priority (PLP) of output queues. This enables you to determine how to prioritize data packets for transmission to the destination. Depending on the priority of the traffic, you can assign packets higher or lower bandwidth, buffer levels, and drop profiles. Based on the conditions defined in the QoS map, packets are forwarded to the next hop.

On hardware vEdge routers and Cloud vEdge virtualized routers, each interface has eight queues, which are numbered 0 to 7. Queue 0 is reserved, and is used for both control traffic and low-latency queuing (LLQ) traffic. For LLQ, any class that is mapped to queue 0 must also be configured to use LLQ. All control traffic is transmitted. Queues 1 to 7 are available for data traffic, and the default scheduling for these seven queues is weighted round-robin (WRR). For these queues, you can define the weighting according to the needs of your network.

Rewrite Data Packets

You can configure and apply rewrite rules on the egress interface to overwrite the Differentiated Services Code Point (DSCP) value for packets entering the network. Rewrite rules allow you to map traffic to code points when the traffic exits the system. Rewrite rules use the forwarding class information and packet loss priority (PLP) used internally by the vEdge routers to establish the DSCP value on outbound packets. You can then configure algorithms such as RED/WRED to set the probability that packets will be dropped based on their DSCP value.

Police Data Packets

You can configure policers to control the maximum rate of traffic sent or received on an interface, and to partition a network into multiple priority levels. Traffic that conforms to the policer rate is transmitted, and traffic that exceeds the policer rate is sent with a decreased priority or is dropped.

You can apply a policer to inbound or outbound interface traffic. Policers applied to inbound interface traffic allow you to conserve resources by dropping traffic that does not need to be routed through the network. Policers applied to outbound interface traffic control the amount of bandwidth used.

Additional Information

Centralized Data Policy
Centralized Data Policy Configuration Examples
Forwarding and QoS Configuration Examples

Forwarding and QoS Configuration Examples

This section shows examples of how you can use access lists to configure quality of service (QoS), classifying data packets and prioritizing the transmission properties for different classes. Note that QoS is synonymous with class of service (CoS).

Forwarding and QoS Example

This example shows how to configure class of service (CoS) to classify data packets and control how traffic flows out of and in to the interfaces on a vEdge router and on the interface queues. To configure a QoS policy:

1. Map each forwarding class to an output queue.
2. Configure the QoS scheduler for each forwarding class.
3. Group the QoS schedulers into a QoS map.
4. Define an access list to specify match conditions for packet transmission.
5. Apply the access list to a specific interface.
6. Apply the queue map and the rewrite rule to the egress interface.

The sections below show examples of each of these steps.

Map Forwarding Class to Output Queue

This example shows a data policy that classifies incoming traffic by mapping each forwarding class to an output queue. Here, traffic classified as "be" (Best Effort) is mapped to queue 2, traffic classified as "af1" (Assured Forwarding) is mapped to queue 3, and so on.

```
policy
 class-map
  class be queue 2
  class af1 queue 3
  class af2 queue 4
  class af3 queue 5
 !
!
```

Configure QoS Scheduler for Each Forwarding Class

This example illustrates how to configure the QoS scheduler for each queue to define the importance of data packets. Depending on the priority of the traffic, you assign the bandwidth, buffer level, and random early detection (RED) drop profile associated with the queue. Here, "af3" traffic has higher priority over other traffic classes and so is configured to have 40% bandwidth and 40% buffer. Traffic in class "af2" has 30% bandwidth and 30% buffer; traffic in class "af1" class has 20% bandwidth and 20% buffer and traffic in class "be" has 10% bandwidth and 10% buffer size reflecting the respective priority of the traffic on the network. All traffic classes are configured with a drop profile of RED, meaning that instead of waiting for the queue to be full, packets are dropped randomly based on the thresholds defined.

```
policy
 qos-scheduler af1
  class             af1
  bandwidth-percent 20
  buffer-percent    20
  drops             red-drop
 !
 qos-scheduler af2
  class             af2
  bandwidth-percent 30
  buffer-percent    30
  drops             red-drop
 !
 qos-scheduler af3
  class             af3
  bandwidth-percent 40
  buffer-percent    40
  drops             red-drop
 !
 qos-scheduler be
  class             be
  bandwidth-percent 10
  buffer-percent    10
  drops             red-drop
 !
```

Group QoS Schedulers into a QoS Map

This example illustrates the grouping of "qos scheduler af1," "qos scheduler af2," and "qos scheduler be" into a single QoS map called "test."

```
qos-map test
  qos-scheduler af1
  qos-scheduler af2
  qos-scheduler be
```

```
 !
!
```

Classify Data Packets into Appropriate Class

This example shows how to classify data packets into appropriate forwarding classes based on match conditions. Here "access-list acl1" classifies data packets originating from the host at source address 10.10.10.1 and going to the destination host at 20.20.20.1 into the "be" class. Data packets with a DSCP value of 10 in the IP header field are classified in the "af1" class, TCP packets are classified in the "af3" class, and packets going to destination port 23, which carries Telnet mail traffic, are classified in the "af2" class. All other traffic is dropped.

```
policy
 access-list acl1
  sequence 1
   match
    source-ip      10.10.10.1/32
    destination-ip 20.20.20.1/32
   !
   action accept
    class be
   !
  !
  sequence 2
   match
    dscp 10
   !
   action accept
    class af1
   !
  !
  sequence 3
   match
    protocol 6
   !
   action accept
    class af3
   !
  !
  sequence 4
   match
    destination-port 23
   !
   action accept
    class af2
   !
  !
  default-action drop
 !
!
```

Apply Access List to Specific Interface

This example illustrates how to apply the access list defined above on the input of a service interface. Here "access-list acl1" is applied on the input of interface ge0/4 in VPN 1.

```
vpn 1
 interface ge0/4
  ip address 10.20.24.15/24
  no shutdown
  access-list acl1 in
```

```
 !
!
```

Configure Rewrite Rule

This example shows how to configure the rewrite rule to overwrite the DSCP field of the outer IP header. Here the rewrite rule "transport" overwrites the DSCP value for forwarding classes based on the drop profile. Since all classes are configured with RED drop, they can have one of two profiles: high drop or low drop. The rewrite rule is applied only on the egress interface, so on the way out, packets classified as "af1" and a Packet Loss Priority (PLP) level of low are marked with a DSCP value of 3 in the IP header field, while "af1" packets with a PLP level of high are marked with 4. Similarly, "af2" packets with a PLP level of low are marked with a DSCP value of 5, while "af2" packets with a PLP level of high are marked with 6, and so on.

```
policy
 rewrite-rule transport
  class af1 low dscp 3
  class af1 high dscp 4
  class af2 low dscp 5
  class af2 high dscp 6
  class af3 low dscp 7
  class af3 high dscp 8
  class be low dscp 1
  class be high dscp 2
 !
!
```

Apply the Queue Map and Rewrite Rule on an Interface

This example applies the queue map "test" and the rewrite rule "transport" to the egress interface ge0/0 in VPN 0. (Note that you cannot apply QOS maps to VLAN interfaces, also called subinterfaces.) Queue maps and rewrite rules are applied only on outgoing traffic.

```
vpn 0
 interface ge0/0
  ip address 10.1.15.15/24
  tunnel-interface
   preference 10
   weight     10
   color      lte
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
  qos-map  test
  rewrite-rule transport
 !
!
```

Police Data Packets

This section shows two examples of policing data packets.

The first example illustrates how to configure a policer to rate limit traffic received on an interface. After you configure the policer, include it in an access list. Here "policer p1" is configured to have a maximum traffic rate of 1,000,000 bits per second and a maximum burst-size limit of 15000 bytes. Traffic exceeding these rate limits is dropped. The policer is then included in the access list "acl1," which is configured

to accept all TCP or UDP traffic originating from the host at source 2.2.0.0 and going to the destination host at 10.1.1.0 on port 20 or 100.1.1.0 on port 30. You can use "access-list acl1" on the input or output of the interface to do flow-based policing.

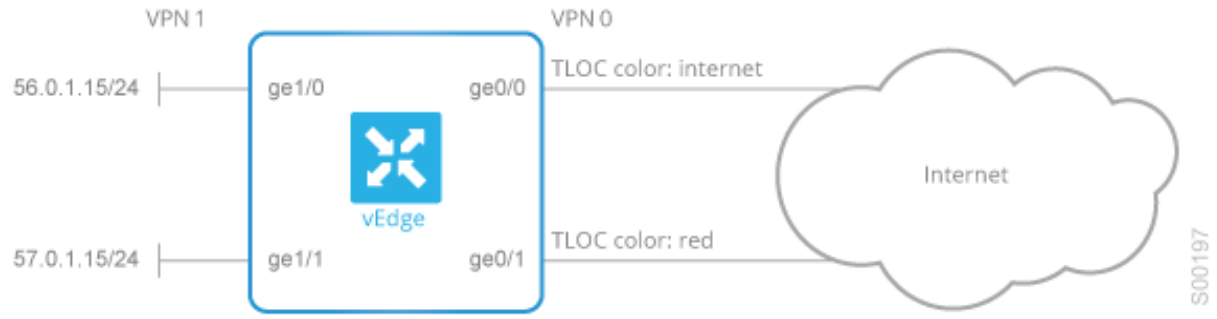
```
policy
 policer p1
  rate   1000000
  burst  15000
  exceed drop
 !
 access-list acl1
  sequence 1
   match
    source-ip        2.2.0.0/16
    destination-ip   10.1.1.0/24 100.1.1.0/24
    destination-port 20 30
    protocol         6 17 23
   !
   action accept
    policer p1
   !
  !
  default-action drop
 !
!
vpn 1
 interface ge0/4
  ip address 10.20.24.15/24
  no shutdown
  access-list acl1 in
 !
!
```

You can also apply a policer directly on an inbound or an outbound interface when you want to police all traffic ingressing or egressing this interface:

```
policy
 policer p1
  rate   1000000
  burst  15000
  exceed drop
 !
!
vpn 1
 interface ge0/4
  ip address 10.20.24.15/24
  no shutdown
  policer p1 in
 !
!

vpn 2
 interface ge0/0
  ip address 10.1.15.15/24
  no shutdown
  policer p1 out
 !
!
```

In the second example, we have a vEdge router with two WAN interfaces in VPN 0. The ge0/0 interface connects to a 30-MB link, and we want to always have 10 MB available for very high priority traffic. When lower-priority traffic bursts exceed 20 MB, we want to redirect that traffic to the second WAN interface, ge0/1.



Implementing this traffic redirection requires two policies:

- You apply an access list to the service-side interface that polices the incoming data traffic.
- You apply a data policy to the ge0/0 WAN interface that directs bursty traffic to the second WAN interface, ge0/1.

For the access list, the configuration snippet belows if for interface ge1/0, in VPN 1. The policer monitors incoming traffic on the interface. When traffic exceeds 20 MB (configured in the **policer burst** command), we change the PLP from low to high (configured by the **policer exceed remark** command). You configure the following on the vEdge router:

```
policy
  policer bursty-traffic
    rate 1000000
    burst 20000
    exceed remark
  access-list policer-bursty-traffic
    sequence 10
      match
        source-ip 56.0.1.0/24
      action accept
        policer bursty-traffic
    default-action accept
vpn 1
  interface ge1/0
    ip address 56.0.1.14/24
    no shutdown
    access-list policer-bursty-traffic in
```

To display a count of the packets that have been remarked, issue the **show interface detail** or the **show system statistics** command on the vEdge router. The count is reported in the rx-policer-remark field.

The centralized data policy directs bursty traffic away from the ge0/0 interface (color: internet) to interface ge0/1 (color: red). You apply this data policy to all the routers at a particular site, specifying the direction **from-service** so that the policy is applied only to traffic originating from the service side of the router. You configure the following on the vSmart controller:

```
policy
  lists
    site-list highest-priority-routers
      site-id 100
    vpn-list wan-vpn
      vpn 0
  data-policy highest-priority
    vpn-list wan-vpn
      sequence 10
        match
          plp high
          source-ip 56.0.1.0/24
        action accept
          count bursty-counter
          set local-tloc color red
```

```
    default-action accept
apply-policy
  site-list highest-priority-routers
    data-policy highest-priority from-service
```

Additional Information

Forwarding and QoS Overview
Localized Data Policy

Forwarding and QoS CLI Reference

CLI commands for configuring and monitoring forwarding and QoS.

Forwarding and QoS Configuration Commands

Use the following commands to configure forwarding and QoS on a vEdge router.

```
policy
  class-map
    class class-name queue number
  cloud-qos
  cloud-qos-service-side
  mirror mirror-name
    remote-dest ip-address source ip-address
  policer policer-name
    rate bandwidth
    burst types
    exceed action
  qos-map map-name
    qos-scheduler scheduler-name
  qos-scheduler scheduler-name
    class class-name
    bandwidth-percent percentage
    buffer-percent percentage
    drops drop-type
  rewrite-rule rule-name
policy
  access-list acl-name
    default-action action
    sequence number
      match
        class class-name
        destination-ip prefix/length
        destination-port number
        dscp number
        protocol number
        source-ip prefix-length
        source-port number
      action
        drop
          count counter-name
        accept
          class class-name
          count counter-name
          mirror mirror-name
          policer policer-name
          set dscp value
vpn vpn-id
  interface interface-name
```

```
    access-list acl-name (in | out)
  interface interface-name
    policer policer-name (in | out)
```

Forwarding and QoS Monitoring Commands

Use the following commands to monitor forwarding and QoS on a vEdge router:

show policy access-list-associations
show policy access-list-counters
show policy access-list-names
show policy access-list-policers
show policy data-policy-filter
show policy qos-map-info
show policy qos-scheduler-info

Additional Information

Forwarding and QoS Overview
Forwarding and QoS Configuration Examples

High Availability and Scaling

High Availability Overview

The goal of any high availability solution is to ensure that all network services are resilient to failure. Such a solution aims to provide continuous access to network resources by addressing the potential causes of downtime through functionality, design, and best practices. The core of the Viptela high availability solution is achieved through a combination of three factors:

- Functional hardware device redundancy. The basic strategy consists of installing and provisioning redundant hardware devices and redundant components on the hardware. These devices are connected by a secure control plane mesh of Datagram Transport Layer Security (DTLS) connections among themselves, which allows for rapid failover should a device fail or otherwise become unavailable. A key feature of the Viptela control plane is that it is established and maintained automatically, by the Viptela devices and software themselves. For more information, see Bringup Sequence of Events .
- Robust network design.
- Software mechanisms ensure rapid recovery from a failure. To provide a resilient control plane, the Viptela Overlay Management Protocol (OMP) regularly monitors the status of all Viptela devices in the network and automatically adjusts to changes in the topology as devices join and leave the network. For data plane resiliency, the Viptela software implements standard protocol mechanisms, specifically Bidirectional Forwarding Detection (BFD), which runs on the secure IPsec tunnels between vEdge routers.

Recovery from a failure is a function of the time it takes to detect the failure and then repair or recover from it. The Viptela solution provides the ability to control the amount of time to detect a failure in the network. In most cases, repair of the failure is fairly instantaneous.

Hardware Support of High Availability

A standard best practice in any network setup is to install redundant hardware at all levels, including duplicate parallel routers and other systems, redundant fans, power supplies and other hardware components within these devices, and backup network connections. Providing high availability in the Viptela solution is no different. A network design that is resilient in the face of hardware failure should include redundant vBond orchestrators, vSmart controllers, and vEdge routers and any available redundant hardware components.

Recovery from the total failure of a hardware component in the Viptela overlay network happens in basically the same way as in any other network. A backup component has been preconfigured, and it is able to perform all necessary functions by itself.

Robust Network Design

In addition to simple duplication of hardware components, the high availability of a Viptela network can be enhanced by following best practices to design a network that is robust in the face of failure. In one such network design, redundant components are spread around the network as much as possible. Design practices include situating redundant vBond orchestrators and vSmart controllers at dispersed geographical locations and connecting them to different transport networks. Similarly, the vEdge routes at a local site can connect to different transport networks and can reach these networks through different NATs and DMZs.

Software Support of High Availability

The Viptela software support for high availability and resiliency in the face of failure is provided both in the control plane, using the standard DTLS protocol and the proprietary Viptela Overlay Management Protocol (OMP), and in the data plane, using the industry-standard protocols BFD, **BGP** , **OSPF** , and VRRP .

Control Plane Software Support of High Availability

Th Viptela control plane operates in conjunction with redundant components to ensure that the overlay network remains resilient if one of the components fails. The control plane is built on top of DTLS connections between the Viptela devices, and it is monitored by the Viptela OMP protocol, which establishes peering sessions (similar to BGP peering sessions) between pairs of vSmart controllers and vEdge routers, and between pairs of vSmart controllers. These peering sessions allow OMP to monitor the status of the Viptela devices and to share the information among them so that each device in the network has a consistent view of the overlay network. The exchange of control plane information over OMP peering sessions is a key piece in the Viptela high availability solution:

- vSmart controllers quickly and automatically learn when a vBond orchestrator or a vEdge router joins or leaves the network. They can then rapidly make the necessary modifications in the route information that they send to the vEdge routers.
- vBond orchestrators quickly and automatically learn when a device joins the network and when a vSmart controller leaves the network. They can then rapidly make the necessary changes to the list of vSmart controller IP addresses that they send to vEdge routers joining the network.
- vBond orchestrators learn when a domain has multiple vSmart controllers and can then provide multiple vSmart controller addresses to vEdge routers joining the network.
- vSmart controllers learn about the presence of other vSmart controllers, and they all automatically synchronize their route tables. If one vSmart controller fails, the remaining systems take over management of the control plane, simply and automatically, and all vEdge routers in the network continue to receive current, consistent routing and TLOC updates from the remaining vSmart controllers.

Let's look at the redundancy provided by each of the Viptela hardware devices in support of network high availability.

vBond Orchestrator Redundancy

The vBond orchestrator performs two key functions in the Viptela overlay network:

- Authenticates and validates all vSmart controllers and vEdge routers that attempt to join the Viptela network.
- Orchestrates the control plane connections between the vSmart controllers and the vEdge routers, thus enabling vSmart controllers and vEdge routers to connect to each other in the Viptela network.

The vBond orchestrator runs as a VM on a network server. The vBond orchestrator can also run on a vEdge router that is configured to be a vBond orchestrator, however this is not recommended, and it limits the number of vEdge router control connections to 50. If using running the vBond daemon on a vEdge router, note that only one vBond daemon can run at a time on a vEdge router, so to provide redundancy and high availability, the network must have two or more vEdge routers that function as vBond orchestrators. (Note also that it is not recommended to use a vEdge router acting as a vBond orchestrator as a regular edge router.)

Having multiple vBond orchestrators ensures that one of them will be always be available whenever a Viptela device—a vEdge router or a vSmart controller—is attempting to join the network.

Configuration of Redundant vBond Orchestrators

A vEdge router learns that it is acting as a vBond orchestrator from its configuration. In the **system vbond** configuration command, which defines the IP address (or addresses) of the vBond orchestrator (or orchestrators) in the Viptela overlay network, you include the **local** option. In this command, you also include the local public IP address of the vBond orchestrator. (Even though on vEdge routers and vSmart controllers you can specify a vBond orchestrator's IP address as a DNS name, on the vBond orchestrator itself you must specify it as an IP address.)

On vSmart controllers and vEdge routers, when the network has only a single vBond orchestrator, you can configure the location of the vBond system either as an IP address or as the name of a DNS server (such as vbond.viptela.com). (Again, you configure this in the **system vbond** command.) When the network has two or more vBond orchestrators and they must all be reachable, you should use the name of a DNS server. The DNS server then resolves the name to a single IP address that the vBond orchestrator returns to the vEdge router. If the DNS name resolves to multiple IP addresses, the vBond orchestrator returns them all to the vEdge router, and the router tries each address sequentially until it forms a successful connection.

Note that even if your Viptela network has only a single vBond orchestrator, it is recommended as a best practice that you specify a DNS name rather than an IP address in the **system vbond** configuration command, because this results in a scalable configuration. Then, if you add additional vBond orchestrators to your network, you do not need to change the configurations on any of the vEdge routers or vSmart controllers in your network.

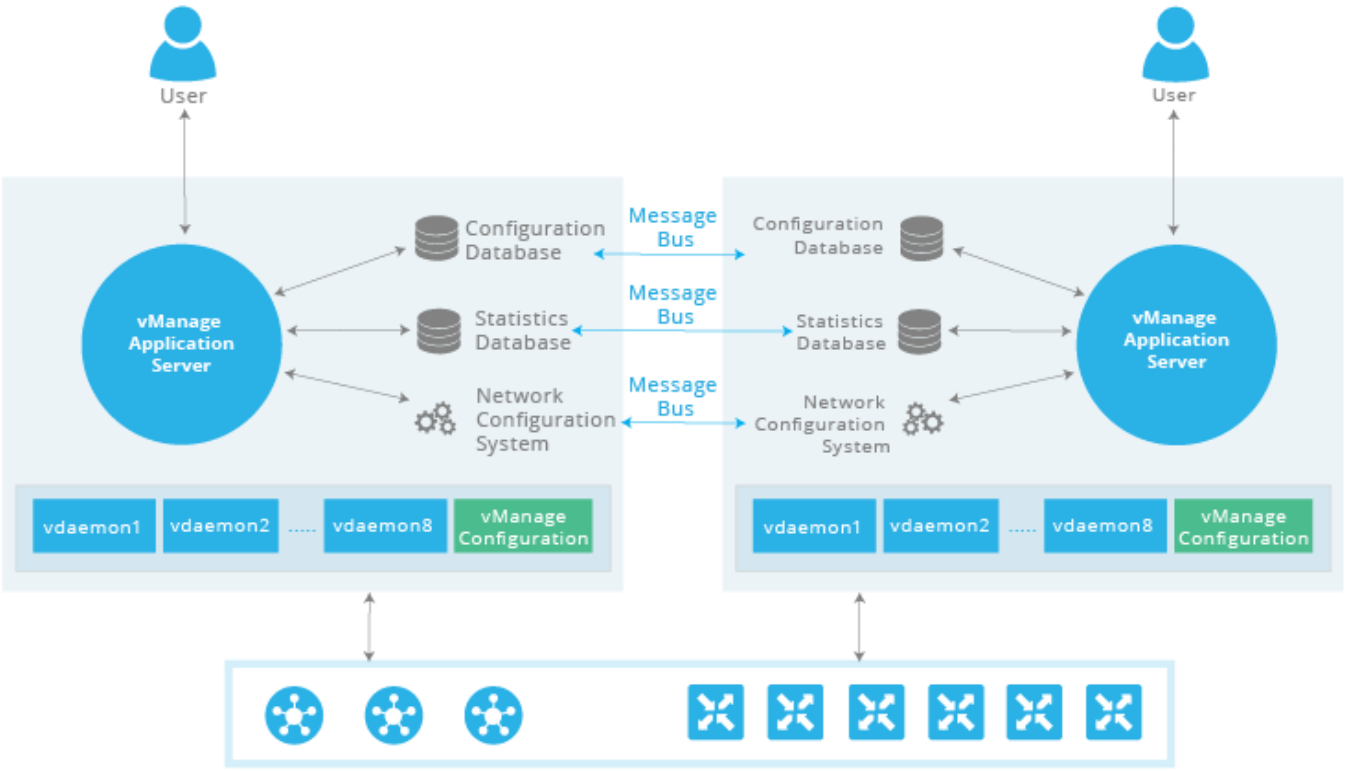
vManage NMS Redundancy

The vManage NMSs comprise a centralized network management system that enables configuration and management of the Viptela devices in the overlay network. It also provides a real-time dashboard into the status of the network and network devices. The vManage NMSs maintain permanent communication channels with all Viptela devices in the network. Over these channels, the vManage NMSs push out files that list the serial numbers of all valid devices, they push out each device's configuration, and they push out new software images as part of a software upgrade process. From each network device, the vManage NMSs receive various status information that is displayed on the vManage Dashboard and other screens.

A highly available Viptela network contains three or more vManage NMSs in each domain. This scenario is referred to as a *cluster* of vManage NMSs, and each vManage NMS in a cluster is referred to as a *vManage instance* . A vManage cluster consists of the following architectural components:

- Application server—This provides a web server for user sessions. Through these sessions, a logged-in user can view a high-level dashboard summary of networks events and status, and can drill down to view details of these events. A user can also manage network serial number files, certificates, software upgrades, device reboots, and configuration of the vManage cluster itself from the vManage application server.
- Configuration database—Stores the inventory and state and the configurations for all Viptela devices.
- Network configuration system—Provides the mechanism for pushing configurations from the vManage NMS to the Viptela devices and for retrieving the running configurations from these devices.
- Statistics database—Stores the statistics information collected from all Viptela devices in the overlay network.
- Message bus—Communication bus among the different vManage instances. This bus is used to share data and coordinate operations among the vManage instances in the cluster.

The following figure illustrates these architectural components. Also in this figure, you see that each vManage instance resides on a virtual machine (VM). The VM can have from one to eight cores, with a Viptela software process (vdaemon) running on each core. In addition, the VM stores the actual configuration for the vManage NMS itself.



The vManage cluster implements an active-active architecture in the following way:

- Each vManage instance in the cluster is an independent processing node.
- All vManage instances are active simultaneously.
- All user sessions to the application server are load-balanced, using either an internal vManage load balancer or an external load balancer.
- All control sessions between the vManage application servers and the vEdge routers are load-balanced. A single vManage instance can manage a maximum of about 2000 vEdge routers. However, all the controller sessions—the sessions between the vManage instances and the vSmart controllers, and the sessions between the vManage instances and the vBond orchestrators—are arranged in a full-mesh topology.
- The configuration and statistics databases can replicated across vManage instances, and these databases can be accessed and used by all the vManage instances.
- If one of the vManage instances in the cluster fails or otherwise becomes unavailable, the network management services that are provided by the vManage NMS are still fully available across the network.

The message bus among the vManage instances in the cluster allows all the instances to communicate using an out-of-band network. This design, which leverages a third vNIC on the vManage VM, avoids using WAN bandwidth for management traffic.

You configure the vManage cluster from the vManage web application server. See the Cluster Manage help article. During the configuration process, you can configure each vManage instance can run the following services:

- Application server—It is recommended that you configure each vManage instance to run an application web servers so that users are able to log in to each of the vManage systems.
- Configuration database—Within the vManage cluster, no more than three iterations of the configuration database can run.

- Load balancer—The vManage cluster requires a load balancer to distribute user login sessions among the vManage instances in the cluster. As mentioned above, a single vManage instance can manage a maximum of 2000 vEdge routers. It is recommended that you use an external load balancer. However, if you choose to use a vManage instance for this purpose, all HTTP and HTTPS traffic directed to its IP address is redirected to other instances in the cluster.
- Messaging server—It is recommended that you configure each vManage instance to run the message bus so that all the instances in the cluster can communicate with each other.
- Statistics database—Within the vManage cluster, no more than three iterations of the statistics database can run.

The following are the design considerations for a vManage cluster:

- A vManage cluster should consist of a minimum of three vManage instances.
- The application server and message bus should run on all vManage instances.
- Within a cluster, a maximum of three instances of the configuration database and three instances of the statistics database can run. Note, however, that any individual vManage instance can run both, one, or none of these two databases.
- To provide the greatest availability, it is recommended that you run the configuration and statistics databases on three vManage instances.

vSmart Controller Redundancy

The vSmart controllers are the central orchestrators of the control plane. They have permanent communication channels with all the Viptela devices in the network. Over the DTLS connections between the vSmart controllers and vBond orchestrators and between pairs of vSmart controllers, the devices regularly exchange their views of the network, to ensure that their route tables remain synchronized. Over DTLS connections to vEdge routers, the vSmart controllers pass accurate and timely route information.

A highly available Viptela network contains two or more vSmart controllers in each domain. A Viptela domain can have up to 20 vSmart controllers, and each vEdge router, by default, connects to two of them. When the number of vSmart controllers in a domain is greater than the maximum number of controllers that a domain's vEdge routers are allowed to connect to, the Viptela software load-balances the connections among the available vSmart controllers.

While the configurations on all the vSmart controllers must be functionally similar, the control policies must be identical. This is required to ensure that, at any time, all vEdge routers receive consistent views of the network. If the control policies are not absolutely identical, different vSmart controllers might give different information to a vEdge router, and the likely result will be network connectivity issues.

To reiterate, the Viptela overlay network works properly only when the control policies on all vSmart controllers are identical. Even the slightest difference in the policies will result in issues with the functioning of the network.

To remain synchronized with each other, the vSmart controllers establish a full mesh of DTLS control connections, as well as a full mesh of OMP sessions, between themselves. Over the OMP sessions, the vSmart controllers advertise routes, TLOCs, services, policies, and encryption keys. It is this exchange of information that allows the vSmart controllers to remain synchronized.

You can place vSmart controllers anywhere in the network. For availability, it is highly recommended that the vSmart controllers be geographically dispersed.

Each vSmart controller establishes a permanent DTLS connection to each of the vBond orchestrators. These connections allow the vBond orchestrators to track which vSmart controllers are present and operational. So, if one of the vSmart controller fails, the vBond orchestrator will not provide the address of the unavailable vSmart controller to a vEdge router that is just joining the network.

vEdge Router Redundancy

vEdge routers are commonly used in two ways in the Viptela network: to be the Viptela routers at a branch site, and to create a hub site that branch vEdge routers connect to.

A branch site can have two vEdge routers in a branch site for redundancy. Each of the router maintains:

- A secure control plane connection, via a DTLS connection, with each vSmart controller in its domain
- A secure data plane connection with the other vEdge routers at the site

Because both vEdge routers receive the same routing information from the vSmart controllers, each one is able to continue to route traffic if one should fail, even if they are connected to different transport providers.

When using vEdge routers in a hub site, you can provide redundancy by installing two vEdge routers. The branch vEdge routers need to connect to each of the hub vEdge routers, using separate DTLS connections.

You can also have vEdge routers provide redundancy by configuring up to tunnel interfaces on a single router. Each tunnel interface can go through the same of different firewalls, service providers, and network clouds, and each maintains a secure control plane connection, by means of a DTLS tunnel, with the vSmart controllers in its domain.

Recovering from a Failure in the Control Plane

The combination of hardware component redundancy with the architecture of the Viptela control plane results in a highly available network, one that continues to operate normally and without interruption when a failure occurs in one of the redundant control plane components. Recovery from the total failure of a vSmart controller, vBond orchestrator, or vEdge router in the Viptela overlay network happens in basically the same way as the recovery from the failure of a regular router or server on the network: A preconfigured backup component is able to perform all necessary functions by itself.

In the Viptela solution, when a network device fails and a redundant device is present, network operation continues without interruption. This is true for all Viptela devices—vBond orchestrators, vSmart controllers, and vEdge routers. No user configuration is required to implement this behavior; it happens automatically. The OMP peering sessions running between Viptela devices ensure that all the devices have a current and accurate view of the network topology.

Let's examine failure recovery device by device.

Recovering from a vSmart Controller Failure

The vSmart controllers are the primary controllers of the network. To maintain this control, they maintain permanent DTLS connections to all the vBond orchestrators and vEdge routers. These connections allow the vSmart controllers to be constantly aware of any changes in the network topology. When a network has multiple vSmart controllers:

- There is a full mesh of OMP sessions among the vSmart controllers.
- Each vSmart controller has a permanent DTLS connection to each vBond orchestrator.
- The vSmart controllers have permanent DTLS connections to the vEdge routers. More specifically, each vEdge router has a DTLS connection to one of the vSmart controllers.

If one of the vSmart controllers fails, the other vSmart controllers seamlessly take over handling control of the network. The remaining vSmart controllers are able to work with vEdge routers joining the network and are able to continue sending route updates to the vEdge routers. As long as one vSmart controller is present and operating in the domain, the Viptela network can continue operating without interruption.

Recovering from a vBond Orchestrator Failure

In a network with multiple vBond orchestrators, if one of them fails, the other vBond orchestrators simply continue operating and are able to handle all requests by Viptela devices to join the network. From a control plane point of view, each vBond orchestrator maintains a permanent DTLS connections to each of the vSmart controllers in the network. (Note however, that there are no connections between the

vBond orchestrators themselves.) As long as one vBond orchestrator is present in the domain, the Viptela network is able to continue operating without interruption, because vSmart controllers and vEdge routers are still able to locate each other and join the network.

Because vBond orchestrators never participate in the data plane of the overlay network, the failure of any vBond orchestrator has no impact on data traffic. vBond orchestrators communicate with vEdge routers only when the routers are first joining the network. The joining vEdge router establishes a transient DTLS connection to a vBond orchestrator to learn the IP address of a vSmart controller. When the vEdge configuration lists the vBond address as a DNS name, the router tries each of the vBond orchestrators in the list, one by one, until it is able to establish a DTLS connection. This mechanism allows a vEdge router to always be able to join the network, even after one of a group of vBond orchestrators has failed.

Recovering from a vEdge Router Failure

The route tables on vEdge routers are populated by OMP routes received from the vSmart controllers. For a site or branch with redundant vEdge routers, the route tables on both remain synchronized, so if either of the routers fails, the other one continues to be able to route data traffic to its destination.

Data Plane Software Support for High Availability

For data plane resiliency, the Viptela software implements the standard BFD protocol, which runs automatically on the secure IPsec connections between vEdge routers. These IPsec connections are used for the data plane, and for data traffic, and are independent of the DTLS tunnels used by the control plane. BFD is used to detect connection failures between the routers. It measures data loss and latency on the data tunnel to determine the status of the devices at either end of the connection.

BFD is enabled, by default, on connections between vEdge routers. BFD sends Hello packets periodically (by default, every 1 second) to determine whether the session is still operational. If a certain number of the Hello packets are not received, BFD considers that the link has failed and brings the BFD session down (the default dead time is 3 seconds). When a BFD sessions goes down, any route that points to a next hop over that IPsec tunnel is removed from the forwarding table (FIB), but it is still present in the route table (RIB).

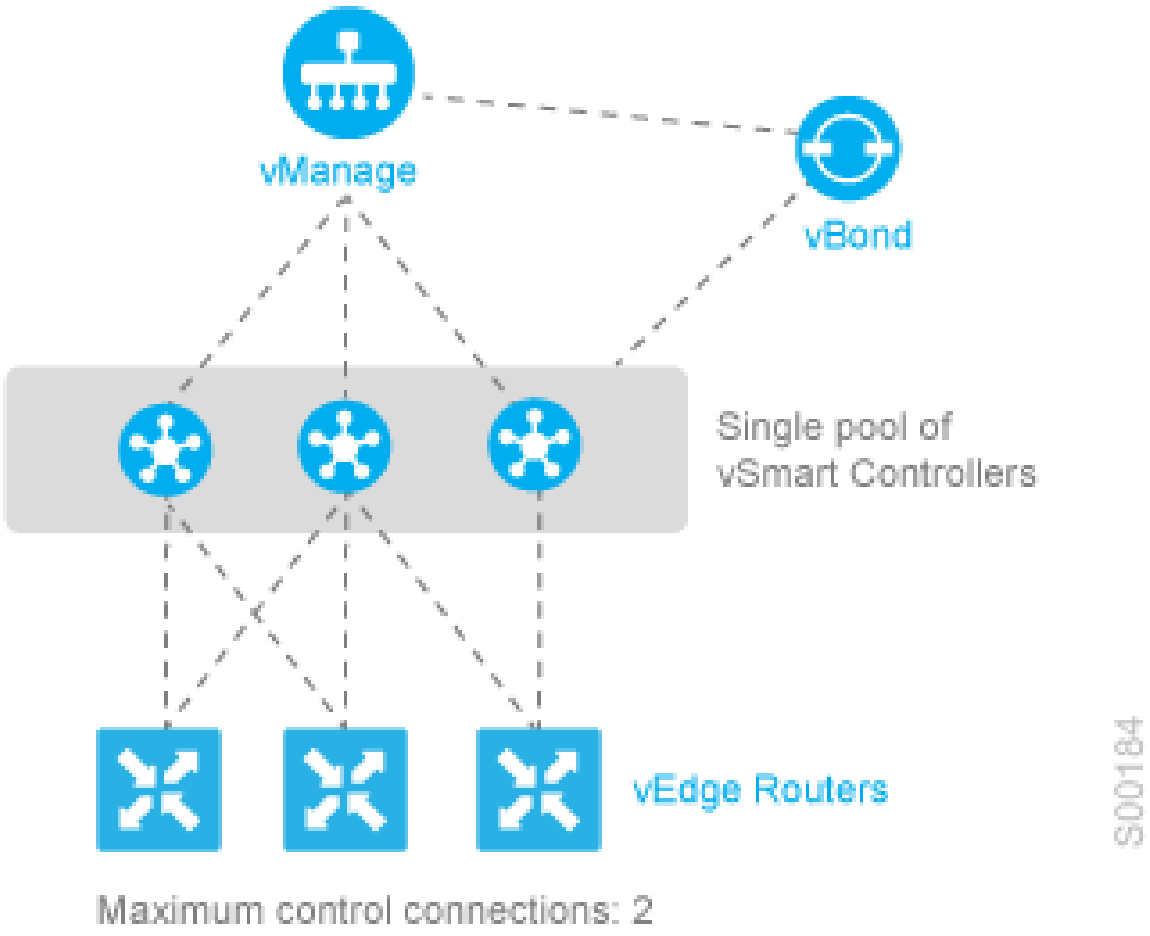
In the Viptela software, you can adjust the Hello packet and dead time intervals. If the timers on the two ends of a BFD link are different, BFD negotiates to use the lower value.

Using Affinity To Manage Network Scaling

In the Viptela overlay network, all vEdge routers establish control connections to all vSmart controllers, to ensure that the routers are always able to properly route data traffic across the network. As networks increase in size, with vEdge routers at thousands of sites and with vSmart controllers in multiple data centers managing the flow of control and data traffic among routers, network operation can be improved by limiting the number of vSmart controllers that a vEdge router can connect to. When data centers are distributed across a broad geography, network operation can also be better managed by having routers establish control connections only with the vSmart controllers collocated in the same geographic region.

Establishing affinity between vSmart controllers and vEdge routers allows you to control the scaling of the overlay network, by limiting the number of vSmart controllers that a vEdge router can establish control connections (and form TLOCs) with. When you have redundant routers in a single data center, affinity allows you to distribute the vEdge control connections across the vSmart controllers. Similarly, when you have multiple data centers in the overlay network, affinity allows you to distribute the vEdge control connections across the data centers. With affinity, you can also define primary and backup control connections, to maintain overlay network operation in case the connection to a single vSmart controller or to a single data center fails.

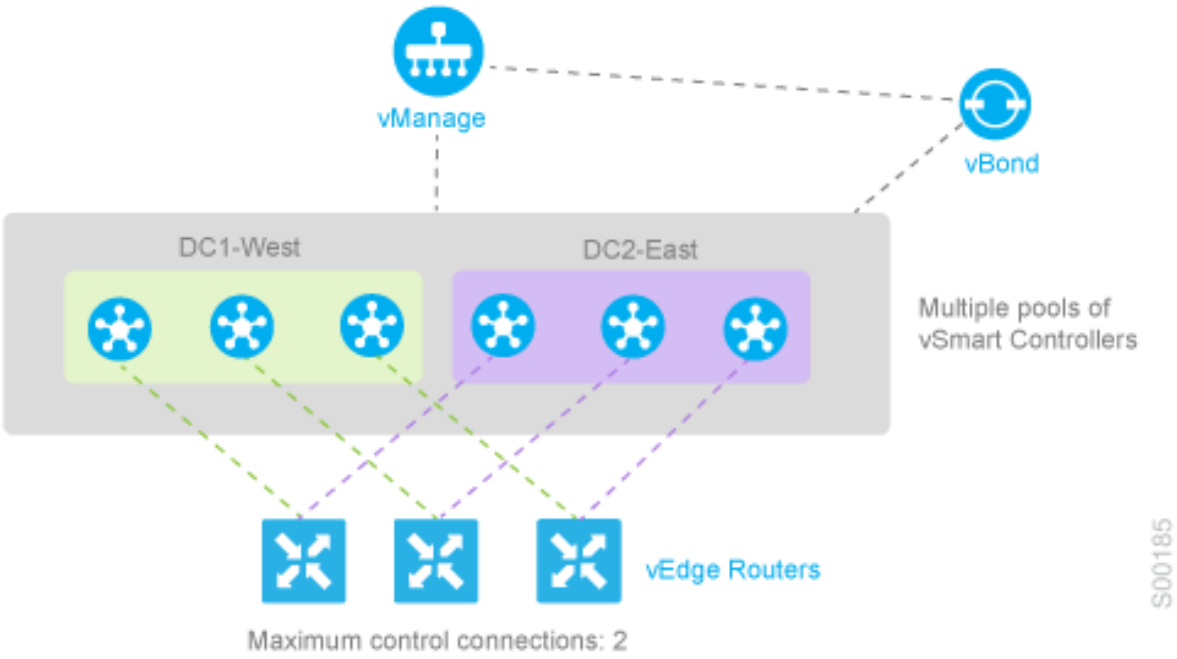
A simple case for using affinity is when redundant vSmart controllers are collocated in a single data center. Together, these vSmart controllers service all the vEdge routers in the overlay network. The figure below illustrates this situation, showing a scenario with three vSmart controllers in the data center and, for simplicity, showing just three of the many vEdge routers in the network.



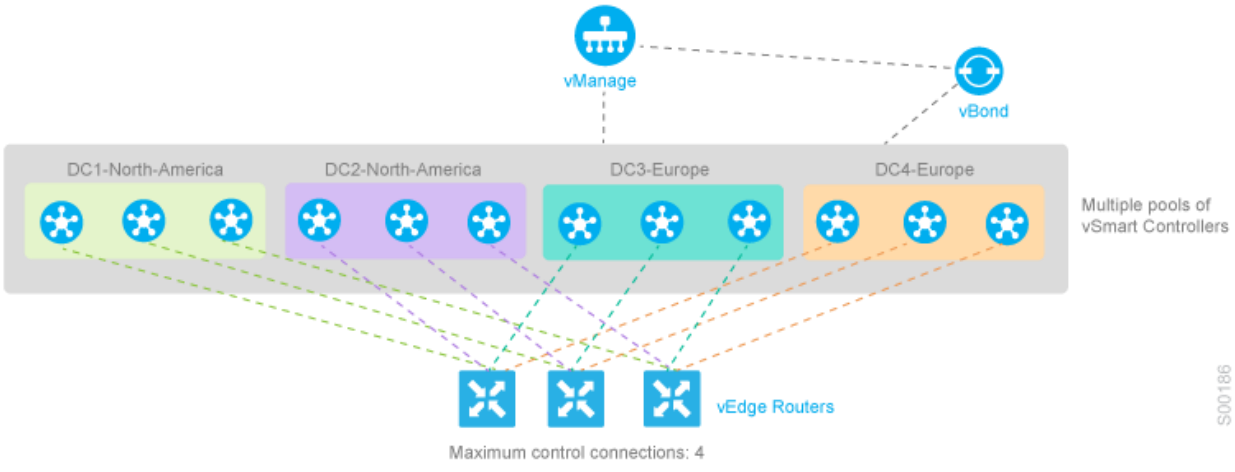
If you do not enable affinity, each vEdge router establishes a control connection—that is, a TLOC—to each of the three vSmart controllers in the data center. Thus, a total of nine TLOCs are established, and each router exchanges OMP updates with each controller. Having this many TLOCs can strain the resources of both the vSmart controllers and the vEdge routers, and the strain increases in networks with larger numbers of vEdge routers.

Enabling affinity allows each vEdge router to establish TLOC connections with only a subset of the vSmart controllers. The figure above shows each router connecting to just two of the three vSmart controllers, thus reducing the total number of TLOCs from nine to six. Both TLOC connections can be active, for a total of six control connections. It is also possible for one of the TLOC connections be the primary, or preferred, and the other to be a backup, to be used as an alternate only when the primary is unavailable, thus reducing the number of active TLOCs to three.

Affinity also enables redundancy among data centers, for a scenario in which multiple vSmart controllers are collocated in two or more data centers. Then, if the link between a vEdge router and one of the data centers goes down, the vSmart controllers in the second data center are available to continue servicing the overlay network. The figure below illustrates this scenario, showing three vSmart controllers in each of two data centers. Each of the three vEdge routers establishes a TLOC connection to one controller in the West data center and one in the East data center.



You might think of the scenario in the figure above as one where there are redundant data centers in the same region of the world, such as in the same city, province, or country. For an overlay network that spans a larger geography, such as across a continent or across multiple continents, you can use affinity to limit the network scale either by restricting vEdge routers so that they connect only to local vSmart controllers or by having vEdge routers preferentially establish control connections with data centers that are in their geographic region. With geographic affinity, vEdge routers establish their only or their primary TLOC connection or connections with vSmart controllers in more local data centers, but they have a backup available to a more distant region to provide redundancy in case the closer data centers become unavailable. The figure below illustrates this scenario, Here, the vEdge routers in Europe have their primary TLOC connections to the two European data centers and alternate connections to the data centers in North America. Similarly, for the vEdge routers in North America, the primary connections are to the two North American data centers, and the backup connections are to the two European data centers.



As is the case with any overlay network that has multiple vSmart controllers, all policy configurations on all the vSmart controllers must be the same.

Additional Information

Configuring Control Plane and Data Plane High Availability Parameters

This article discusses the configurable high availability parameters for the control plane and the data plane.

Control Plane High Availability

A highly available Viptela network contains two or more vSmart controllers in each domain. A Viptela domain can have up to 20 vSmart controllers, and each vEdge router, by default, connects to two of them. You change this value on a per-tunnel basis:

```
vEdge(config-tunnel-interface)# max-control-connections number
```

When the number of vSmart controllers in a domain is greater than the maximum number of controllers that a domain's vEdge routers are allowed to connect to, the Viptela software load-balances the connections among the available vSmart controllers.

To maximize the efficiency of the load-balancing among vSmart controllers, use sequential numbers when assigning system IP addresses to the vEdge routers in the domain. One example of a sequential numbering schemes is 172.1.1.1, 172.1.1.2, 172.1.1.3, and so forth. Another is 172.1.1.1, 172.1.2.1, 172.1.3.1, and so forth.

Data Plane High Availability

BFD, which detects link failures as part of the Viptela high availability solution, is enabled by default on all Viptela devices. BFD runs automatically on all IPsec data tunnels between vEdge routers. It does not run on the control plane (DTLS or TLS) tunnels that vSmart controllers establish with all Viptela devices in the network.

You can modify the BFD Hello packet interval and the number of missed Hello packets (the BFD interval multiplier) before BFD declares that a link has failed.

Change the BFD Hello Packet Interval

BFD sends Hello packets periodically to detect faults on the IPsec data tunnel between two vEdge routers. By default, BFD sends these packets every 1000 milliseconds (that is, once per second). To change this interval on one or more traffic flow, use the **hello-interval** command:

```
vEdge(config)# bfd color color hello-interval milliseconds
```

The interval can be a value from 100 to 300000 milliseconds (5 minutes).

You configure the interval for each tunnel connection, which is identified by a color. The color can be **3g** , **biz-internet** , **blue** , **bronze** , **custom1** , **custom2** , **custom3** , **default** , **gold** , **green** , **lte** , **metro-ethernet** , **mpls** , **private1** , **private2** , **public-internet** , **red** , or **silver** .

Change the BFD Packet Interval Multiplier

After BFD has not received a certain number of Hello packets on a link, it declares that the link has failed. This number of packets is a multiplier of the Hello packet interval time. By default, the multiplier is 7 for hardware vEdge routers and 20 for vEdge Cloud software

routers. This means that if BFD has not received a Hello packet after 7 seconds, it considers that the link has failed and implements its redundancy plan.

To change the BFD packet interval multiplier, use the **multiplier** command:

```
vEdge(config)# bfd color color multiplier integer
```

The multiplier can be an integer from 1 to 60.

You configure the multiplier for each tunnel connection, which is represented by a color.

Control PMTU Discovery

On each transport connection (that is, for each TLOC, or color), the Viptela BFD software performs path MTU (PMTU) discovery, which automatically negotiates the MTU size in an effort to minimize or eliminate packet fragmentation on the connection. BFD PMTU discovery is enabled by default, and it is recommended that you use BFD PMTU discovery and not disable it. To explicitly enable it:

```
vEdge(config)# bfd color color pmtu-discovery
```

With PMTU discovery enabled, the path MTU for the tunnel connection is checked periodically, about once per minute, and it is updated dynamically. With PMTU discovery enabled, 16 bytes might be required by PMTU discovery, so the effective tunnel MTU might be as low as 1452 bytes. From an encapsulation point of view, the default IP MTU for GRE is 1468 bytes, and for IPsec it is 1442 bytes because of the larger overhead. Enabling PMTU discovery adds to the overhead of the BFD packets that are sent between the vEdge routers, but does not add any overhead to normal data traffic.

If PMTU discovery is disabled, the expected tunnel MTU is 1472 bytes (tunnel MTU of 1500 bytes less 4 bytes for the GRE header, 20 bytes for the outer IP header, and 4 bytes for the MPLS header). However, the effective tunnel MTU might be 1468 bytes, because the software might sometimes erroneously add 4 bytes to the header.

Additional Information

High Availability Overview

Configuring Affinity between vSmart and vEdge Devices

One way to manage network scale is to configure affinity between vSmart controllers and vEdge routers. To do this, you place each vSmart controller into a controller group, and then you configure which group or groups a vEdge router can establish control connections with. The controller groups are what establishes the affinity between vSmart controllers and vEdge routers.

Configure the Controller Group Identifier on vSmart Controllers

To participate in affinity, each vSmart controller must be assigned a controller group identifier:

```
vSmart(config#) system controller-group-id number
```

The identifier number can be from 0 through 100.

When vSmart controllers are in different data centers, it is recommended that you assign different controller group identifiers to the vSmart controllers. Doing this provides redundancy among data centers, in case a data center becomes unreachable.

For vSmart controllers in the same a data center, they can have the same controller group identifier or different identifiers:

- If the vSmart controllers have the same controller group identifier, a vEdge router establishes a control connection to any one of them. If that vSmart controller becomes unreachable, the router simply establishes a control connection with another one of the controllers in the data center. As an example of how this might work, if one vSmart controller becomes unavailable during a software upgrade, the vEdge router immediately establishes a new TLOC with another vSmart controller, and the router's network operation is not interrupted. This network design provides redundancy among vSmart controllers in a data center.

- If the vSmart controllers have different controller group identifiers, a vEdge router can use one controller as the preferred and the other as backup. As an example of how this might work, if you are upgrading the vSmart controller software, you can upgrade one controller group at a time. If a problem occurs with the upgrade, a vEdge router establishes TLOCs with the vSmart controllers in the second, backup controller group, and the router's network operation is not interrupted. When the vSmart controller in the first group again become available, the vEdge router switches its TLOCs back to that controller. This network design, while offerring redundancy among the vSmart controllers in a data center, also provides additional fault isolation.

Configure Affinity on vEdge Routers

For a vEdge router to participate in affinity, you configure the vSmart controllers that the router is allowed to establish control connections with, and you configure the maximum number of control connections (or TLOCs) that the vEdge router itself, and that an individual tunnel on the router, is allowed to establish.

Configure a Controller Group List

Configuring the vSmart controllers that the router is allowed to establish control connections is a two-part process:

- At the system level, configure a single list of all the controller group identifiers that are present in the overlay network.
- For each tunnel interface in VPN 0, you can choose to restrict which controller group identifiers the tunnel interface can establish control connections with. To do this, configure an exclusion list.

At a system level, configure the identifiers of the vSmart controller groups:

```
vEdge(config)# system controller-group-list numbers
```

List the vSmart controller group identifiers that any of the tunnel connections on the vEdge router might want to establish control connections with. It is recommended that this list contain the identifiers for all the vSmart controller groups in the overlay network.

If, for a specific tunnel interface in VPN 0, you want it to establish control connections to only a subset of all the vSmart controller groups, configure the group identifiers to exclude:

```
vEdge(config-vpn-0-interface)# tunnel-interface exclude-controller-group-list numbers
```

In this command, list the identifiers of the vSmart controller groups that this particular tunnel interface should never establish control connections with. The controller groups in this list must be a subset of the controller groups configured with the **system controller-group-list** command.

To display the controller groups configured on a vEdge router, use the **show control connections** command.

Configure the Maximum Number of Control Connections

Configuring the maximum number of control connections for the vEdge router is a two-part process:

- At the system level, configure that maximum number of control connections that the vEdge router can establish to vSmart controllers.
- For each tunnel interface in VPN 0, configure the maximum number of control connections that the tunnel can establish to vSmart controllers.

By default, a vEdge router can establish two OMP sessions for control connections to vSmart controllers. To modify the maximum number of OMP sessions:

```
vEdge(config)# system max-omp-sessions number
```

The number of OMP sessions can be from 0 through 100.

A vEdge router establishes OMP sessions as follows:

- Each DTLS and and each TLS control plane tunnel creates a separate OMP session.
- It is the vEdge router as a whole, not the individual tunnel interfaces in VPN 0, that establishes OMP sessions with vSmart controllers. When different tunnel interfaces on the router have affinity with the same vSmart controller group, the vEdge router creates a single OMP session to one of the vSmart controllers in that group, and the different tunnel interfaces use this single OMP session.

By default, each tunnel interface in VPN 0 can establish two control connections. To change this:

```
vEdge(config-vpn-0-interface)# vpn 0 interface interface-name tunnel-interface max-control-connections
number
```

The number of control connections can be from 0 through 100. The default value is the maximum number of OMP sessions configured with the **system max-omp-sessions** command.

When a vEdge routers has multiple WAN transport connections, and hence has multiple tunnel interfaces in VPN 0, the sum of the maximum number of control connections that all the tunnels can establish cannot exceed the maximum number allowed on the router itself.

To display the maximum number of control connections configured on an interface, use the **show control local-properties** command.

To display the actual number of control connections for each tunnel interface, use the **show control affinity config** command.

To display a list of the vSmart controllers that each tunnel interface has established control connections with, use the **show control affinity status** command.

Best Practices for Configuring Affinity

- In the **system controller-group-list** command on the vEdge router, list all the controller groups available in the overlay network. Doing so ensures that all the vSmart controllers in the overlay network are available for the affinity configuration, and it provides additional redundancy in case connectivity to the preferred group or groups is lost. You manipulate the number of control connections and their priority based on the maximum number of OMP sessions for the router, the maximum number of control connections for the tunnel, the controller groups a tunnel should not use.
  A case in which listing all the controller groups in the **system controller-group-list** command provides additional redundancy is when the vEdge router site is having connectivity issues in reaching the vSmart controllers in the controller group list. To illustrate this, suppose, in a network with three controller groups (1, 2, and 3), the controller group list on a vEdge router contains only groups 1 and 2, because these are the preferred groups. If the router learns from the vBond controller that the vSmart controllers in groups 1 and 2 are up, but the router is having connectivity issues to both sites, the router loses its connectivity to the overlay network. However, if the controller group list contains all three controller groups, even though group 3 is not a preferred group, if the router is unable to connect to the vSmart controllers in group 1 or group 2, it is able to fall back and connect to the controllers in group 3.
  Configuring affinity and the order in which to connect to vSmart controllers is only a preference. The preference is honored whenever possible. However, the overarching rule in enforcing high availability on the overlay network is to use any operational vSmart controller. The network ceases to function only when no vSmart controllers are operational. So it might happen that the least preferred vSmart controller is used if it is the only controller operational in the network at a particular time.
  When a vEdge router boots, it learns about all the vSmart controllers in the overlay network, and the vBond orchestrator is continuously communicating to the router which vSmart controllers are up. So, if a vEdge router cannot reach any of the preferred vSmart controllers in the configured controller group and another vSmart controller is up, the router connects to that controller. Put another way, in a network with multiple vSmart controllers, as a last resort, a vEdge router connects to any of the controllers, to ensure that the overlay network remains operational, whether or not these controllers are configured in the router's controller group list.
- The controller groups listed in the **exclude-controller-group-list** command must be a subset of the controller groups configured for the entire router, in the **system controller-group-list** command.
- When a data center has multiple vSmart controllers that use the same controller group identifier, and when the overlay network has two or more data centers, it is recommended that the number of vSmart controllers in each of the controller groups be the same. For example, if Data Center 1 has three vSmart controllers, all with the same group identifier (let's say, 1), Data Center 2 should also have three vSmart controllers, all with the same group identifier (let's say, 2), and any additional data centers should also have three vSmart controllers.

- When a data center has vSmart controllers in the same controller group, the hardware capabilities—specifically, the memory and CPU—on all the vSmart controllers should be identical. More broadly, all the vSmart controllers in the overlay network, whether in one data center or in many, should have the same hardware capabilities. Each vSmart controller should have equal capacity and capability to handle a control connection from any of the vEdge routers in the network.
- When a router has two tunnel connections and the network has two (or more) data centers, it is recommended that you configure one of the tunnel interfaces to go to one of the data centers and the other to go to the second. This configuration provides vSmart redundancy with the minimum number of OMP sessions.
- Whenever possible in your network design, you should leverage affinity configurations to create fault-isolation domains.

Additional Information

High Availability Configuration Examples
High Availability Overview

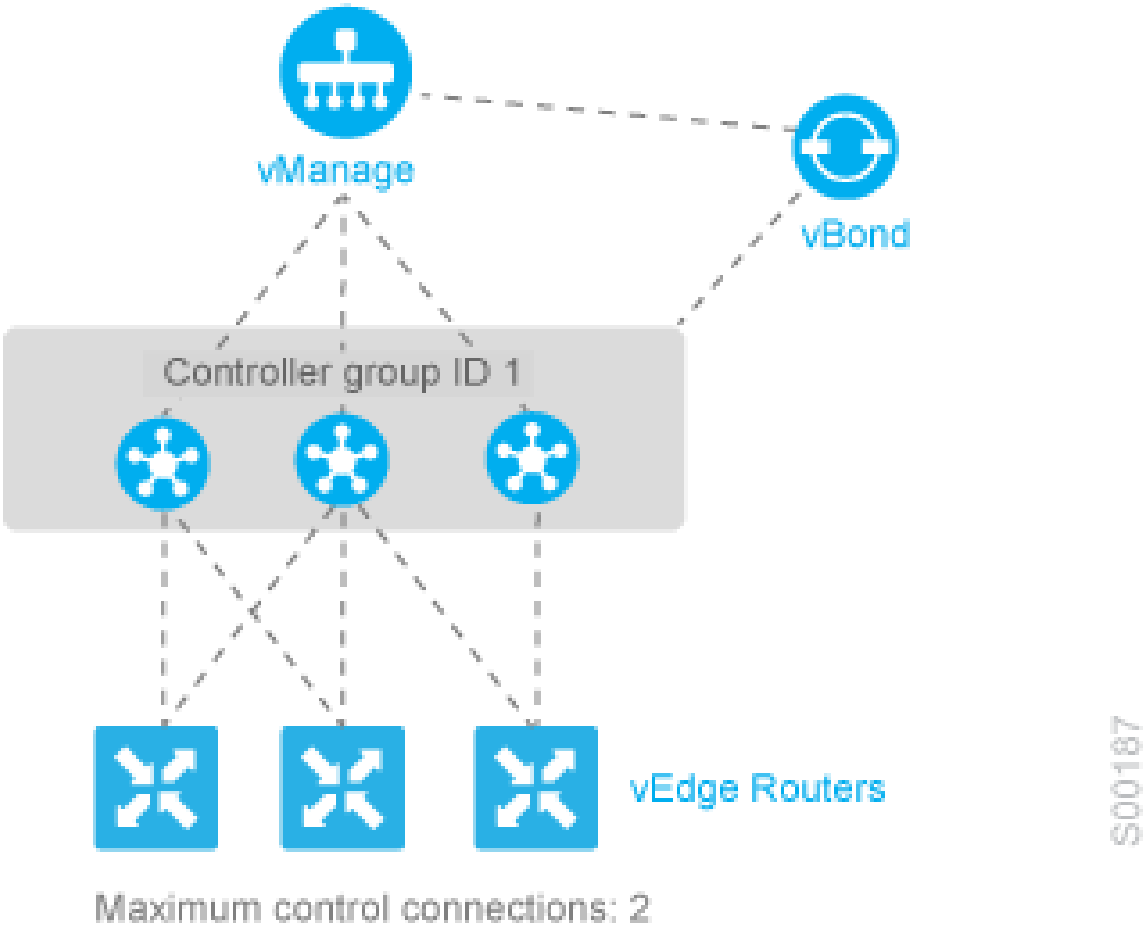
High Availability Configuration Examples

This article provides examples of configuring high availability, specifically, of configuring affinity between vSmart controllers and vEdge routers.

Configure Affinity to vSmart Controllers in a Single Data Center

In an overlay network with a single data center that has multiple vSmart controllers, if you want the vEdge routers to establish a single control connection to one of vSmart controllers, there is no need to configure affinity because this situation is the default behavior.

However, if you want the vEdge routers to establish control connections to more than one vSmart controllers, to provide redundancy in case one of the controllers becomes unavailable, you configure affinity. You generally place the vSmart controllers in the same controller group.



Let's say that all the vSmart controllers use the same controller group identifier, 1. You configure the identifier on all three controllers as follows:

```
vSmart(config)# system controller-group-id 1
```

To verify the configuration, use the **show running-config** command:

```
vSmart# show running-config system
system
 description         "vSmart in data center 1"
 host-name           vSmart
 gps-location latitude 37.368140
 gps-location longitude -121.913658
 system-ip           172.16.255.19
 site-id             100
 controller-group-id 1
 organization-name   "Viptela Inc"
 clock timezone America/Los_Angeles
...
```

We want the three vEdge routers to establish two control connections to two of the three vSmart controllers. We do this for purposes of redundancy, in case one of the controllers becomes available. Because all the vSmart controllers are in the same controller group, we cannot specify or influence which of the two controllers the vEdge routers connect to. The configurations on all three routers are effectively identical. We show here the configuration for router vEdge-1.

First, configure the available vSmart controller groups. This scenario has just one group:

```
vEdge-1(config)# system controller-group-list 1
```

By default, a vEdge router can establish two control connections. Because we want each vEdge router and each tunnel interface to connect to two vSmart controllers, no configuration is required here. However, if you want to explicitly configure the these parameters, you configure the maximum number of OMP sessions at the system level and the maximum number of control connections per tunnel:

```
vEdge-1(config)# system max-omp-sessions 2
vEdge-1(config)# vpn 0 interface ge0/2 tunnel-interface
vEdge-1(config-tunnel-interface)# max-control-connections 2
```

Here are the relevant configuration snippets from vEdge-1:

```
vEdge-1# show running-config system
system
 host-name             vEdge-1
 gps-location latitude 43.0
 gps-location longitude -75.0
 system-ip             172.16.255.11
 site-id               100
 max-omp-sessions      2
 controller-group-list 1
 organization-name     "Viptela Inc"
...
vEdge-1# show running-config vpn 0
...
 interface ge0/2
  ip address 10.0.5.11/24
  tunnel-interface
   encapsulation ipsec
   color lte
   max-control-connections 1
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
  !
  no shutdown
...
```

To display the control connections with the vSmart controllers, use the **show control connections** command. The last column, Controller Group ID, lists the vSmart controller group that the vEdge router is in.

```
vEdge1# show control connections
                                                                                    PEER
PEER                                              CONTROLLER
PEER     PEER     PEER             SITE        DOMAIN      PEER             PRIVATE  PEER
PUBLIC                                            GROUP
TYPE     PROTOCOL SYSTEM IP        ID          ID          PRIVATE IP       PORT     PUBLIC IP
PORT    LOCAL COLOR      STATE           UPTIME     ID
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------
vsmart   dtls     172.16.255.19    100         1           10.0.5.19        12446    10.0.5.19
12446   lte              up              0:00:00:53 1
vsmart   dtls     172.16.255.20    200         1           10.0.12.20       12446    10.0.12.20
12446   lte              up              0:00:00:22 1
```

To display the maximum number of control connections allowed on the router, use the **show control local-properties** command. The last column of the last line of the output lists the maximum controllers. The following is the abbreviated output for this command:

```
vEdge-1# show control local-properties
personality                 vedge
organization-name           Viptela Inc
certificate-status          Installed
root-ca-chain-status        Installed

certificate-validity        Valid
certificate-not-valid-before Mar 10 19:50:04 2016 GMT
certificate-not-valid-after  Mar 10 19:50:04 2017 GMT
...

                  PUBLIC        PUBLIC  PRIVATE       PRIVATE                                  MAX
INDEX  INTERFACE  IP            PORT    IP            PORT     VSMARTS... LAST CONNECTION  CONTROLLERS
-------------------------------------------------------------------------------------------------------
0      ge0/2      10.0.5.11     12386   10.0.5.11     12386    2      ... 0:00:07:54       2
```

Two commands display information about the control connections established by the affinity configuration. To see, for each interface, which controller groups are configured and which the interface is connected to, use the **show control affinity config** command:

```
vEdge-1# show control affinity config

EFFECTIVE CONTROLLER LIST FORMAT - G(C),...     - Where G is the Controller Group ID
                                                  C is the Required vSmart Count
CURRENT CONTROLLER LIST FORMAT   - G(c)s,...    - Where G is the Controller Group ID
                                                  c is the current vSmart count
                                                  s Status ✓ when matches, χ when does not match
               EFFECTIVE
               REQUIRED
INDEX INTERFACE VS COUNT  EFFECTIVE CONTROLLER LIST     CURRENT CONTROLLER LIST           EQUILIBRIUM
-------------------------------------------------------------------------------------------------------
0     ge0/2     2           1(2)                        1(2)✓                             Yes
```

The command output above shows that affinity is configured on interface ge0/2.

- The Effective Required vs. Count column shows that the interface is configured to create two control connections, and, in fact, two control connections have been established. You configure the number of control connections for the tunnel interface with the **max-control-connections** command.
- The Effective Controller List column shows that affinity on the interface is configured to use vSmart controller identifier 1 and that the router supports two OMP sessions. You configure the affinity controller identifiers with the **controller-group-list** command (at the **system** level) and, for the tunnel interface, the **exclude-controller-group-list** command.
- The Current Controller List column lists the actual affinity configuration for the interface. The output here shows that the interface has two control connections with vSmart controllers in group 1. The check mark indicates that the current and effective controller lists match each other. If, for example, the tunnel had established only one TLOC connection to a vSmart controller, this column would show "1(1)X".
- The Equilibrium column indicates that the current controller lists matches what is expected from the affinity configuration for that tunnel interface.

To determine the exact vSmart controllers that the tunnel interface has established control connections with, use the **show control affinity status** command:

```
vEdge-1# show control affinity status

ASSIGNED CONNECTED CONTROLLERS   - System IP( G),..  - System IP of the assigned vSmart
                                                      G is the group ID to which the vSmart
belongs to

UNASSIGNED CONNECTED CONTROLLERS - System IP( G),..  - System IP of the unassigned vSmart
                                                      G is the group ID to which the vSmart
belongs to
```

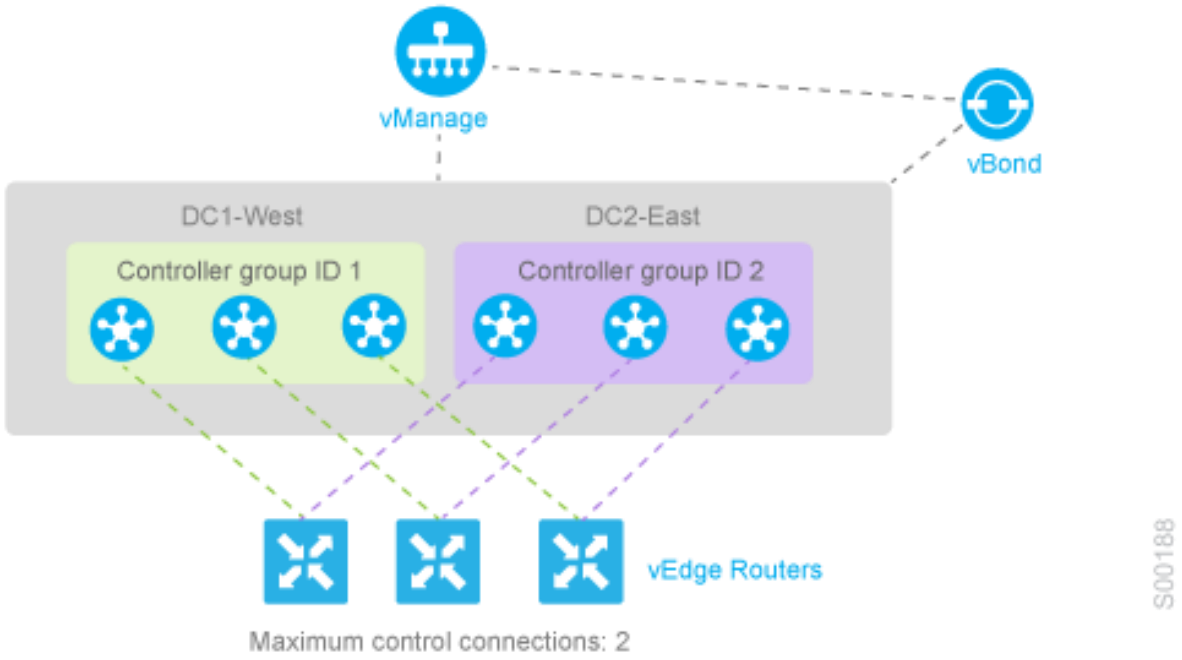
```
INDEX INTERFACE ASSIGNED CONNECTED CONTROLLERS              UNASSIGNED CONNECTED CONTROLLERS
-------------------------------------------------------------------------------------------------------------
---------------
0     ge0/2       172.16.255.19(  1),  172.16.255.20(  1)
```

The command output above shows that interface **ge0/2** has control connections to two vSmart controllers, 172.16.255.19 and 172.16.255.20, that both controllers are in group 1, and that both controllers are in one of the groups configured in the controller group list. If the interface were connected to a vSmart controller not in the controller group list, it would be listed in the Unassigned Connected Controllers column.

When a data center has multiple vSmart controllers, you can configure them to be in different controller groups. For example, if you configure them to be in two different controller groups, you could each vEdge router to establish two control connections, one to each of the groups. While this configuration design is similar to what we discussed in the previous section, providing redundant control connections to the vSmart controllers, on subtle difference is that it provides fault isolation between the two vSmart controller groups in the data center.The configuration for this scenario is almost identical to the configuration when vSmart controllers are two data centers. The only difference is that here, two vSmart controller groups are collocated in the same data center. See the configuration example in the next section.

Configure Affinity to vSmart Controllers in Two Data Centers

You can use affinity to enable redundancy among data centers, for a network design in which multiple vSmart controllers are spread across two or more data centers. Then, if the link between a vEdge router and one of the data centers goes down, the vSmart controllers in the second data center are available to continue servicing the overlay network. The figure below illustrates this scenario, showing three vSmart controllers in each of two data centers. Each of the three vEdge routers establishes a TLOC connection to one controller in the West data center and one in the East data center.



You configure the three vSmart controllers in DC1-West with controller group identifier 1:

```
vSmart-DC1(config)# system controller-group-id 1
```

The three vSmart controllers in DC2-East are in controller group 2:

```
vSmart-DC2(config)# system controller-group-id 2
```

We want all the vEdge routers to have a maximum of two OMP sessions, and we want each tunnel interface to have a maximum of two control connections and to not exclude any controller groups. So the only configuration that needs to be done on the vEdge routers is to set the controller group list. We want vEdge routers in the west to prefer vSmart controllers in DC1-West over DC2-East:

```
vEdge-West(config)# system controller-group-list 1 2
```

Similarly, we want vEdge routers in the east to prefer DC2-East:

```
vEdge-East(config)# system controller-group-list 2 1
```

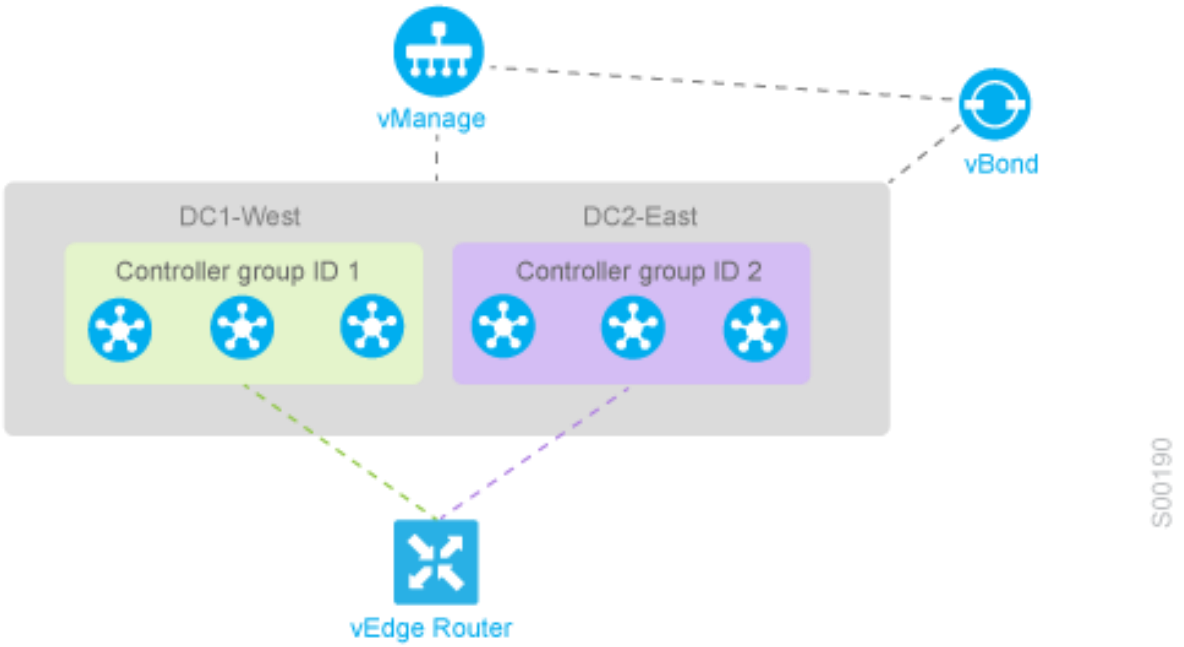
The software evaluates the controller group list in order, so with this configuration, the vEdge-West routers prefer vSmart controller group 1 (which is the West data center), and the vEdge-East routers prefer vSmart controller group 2.

You can fine-tune the controller group preference in other ways:

- Set the maximum number of OMP sessions allowed on the router to 1 (**system max-omp-sessions 1**). To illustrate how this works, let's look at a vEdge-West router. The router has only one tunnel interface, and that interface creates one control connection to vSmart controller list 1. If all the vSmart controllers in this group become unavailable, or if the connection between the router that the DC1-West data center goes down, the tunnel interface establishes one control connection to vSmart controller list 2, because this group is listed in the **system controller-group-list** command. If all vSmart controllers in both controller groups, or the connections to them, become unavailable, and if the vBond orchestrator also indicates that all these vSmart controllers are unreachable, the tunnel interface establishes a control connection to any other vSmart controller in the overlay network if other controllers are present.
- Set the maximum number of control connections that the tunnel interface can establish to 1 (**vpn 0 interface tunnel-interface max-control-connections 1**). Because the software evaluates the controller group list in order, for a vEdge-West router, this configuration forces the tunnel interface to establish a control connection to vSmart controller group 1. Again, if this controller group or data center becomes unreachable, the tunnel establishes a control connection with controller group 2, because this group is configured in the **system controller-group-list** command. And if neither controller group 1 or 2 is available, and if another vSmart controller is present in the network, the tunnel interface establishes a control connection with that controller.
- Exclude the non-preferred vSmart controller group for a particular tunnel. For example, for a vEdge-West router to prefer controller group 1, you configure **vpn 0 interface tunnel-interface exclude-controller-group-list 2** . As with the above configurations, if this controller group or data center becomes unreachable, the tunnel establishes a control connection with controller group 2, because this group is configured in the **system controller-group-list** command. And if neither controller group 1 or 2 is available, and if another vSmart controller is present in the network, the tunnel interface establishes a control connection with that controller.

Configure Redundant Control Connections on One vEdge Router

When a router has two tunnel connections and the network has two (or more) data centers, you can configure redundant control connections from the vEdge router to vSmart controllers in two of the data centers. It is recommended that do this using the minimum number of OMP sessions—in this case, two. To do this, you configure one of the tunnel interfaces to go only to one of the data centers and the other to go only to the second. This configuration provides vSmart redundancy with the minimum number of OMP sessions.



On the vEdge router, define the controller group list and configure the maximum number of OMP sessions to be 2:

```
vEdge(config-system)# controller-group-list 1 2
vEdge(config-system)# max-omp-sessions 2
```

For one of the tunnels, you can use the default affinity configuration (that is, there is nothing to configure) to have this tunnel prefer a vSmart controller in group 1. You can also explicitly force this tunnel to prefer vSmart controller group 1:

```
vEdge(config-tunnel-interface-1)# max-control-connections 1
```

You do not need to configure **exclude-controller-group-list 2** , because the software evaluates the controller group list in order, starting with group 1. However, you could choose to explicitly exclude vSmart controller group 2.

Then, on the second tunnel, configure it to prefer a vSmart controller in group 2. As with the other tunnel, you limit the maximum number of control connections to 1. In addition, you have to exclude controller group 1 for this tunnel.

```
vEdge(config-tunnel-interface-2)# max-control-connections 1
vEdge(config-tunnel-interface-2)# exclude-controller-group-list 1
```

Additional Information

High Availability Overview
Using Affinity to Manage Network Scale

High Availability CLI Reference

CLI commands for configuring and monitoring high availability.

High Availability Configuration Commands

Use the following commands to configure high availability on a vEdge router:

```
bfd
  app-route
```

```
    multiplier number
    poll-interval milliseconds
  color color
    hello-interval milliseconds
    multiplier number
    pmtu-discovery
```

High Availability Monitoring Commands

show bfd sessions —Display information about the BFD sessions running on the local vEdge router.

Additional Information

Configuring High Availability Parameters
High Availability Overview

Network Optimization

Network Optimization Overview

The Viptela network optimization solution includes the following:

- CloudExpress Service
- TCP Optimization

CloudExpress Service

Enterprise software providers deliver many applications as Software as a Service (SaaS) cloud applications, such as Dropbox, Microsoft Office365, and Salesforce. Latency and packet loss impact the performance of these applications, but in legacy networks, network administrators have little visibility into network characteristics between end users and SaaS applications. When a path is impaired in a legacy network, the manual process of shifting application traffic to an alternate path is complex, time consuming, and error prone.

CloudExpress service addresses these issues by optimizing performance for SaaS applications in the Viptela overlay network. From a central dashboard, CloudExpress service provides clear visibility into the performance of individual cloud applications and automatically chooses the best path for each one. It responds to changes in network performance in real-time, intelligently re-routing cloud application traffic onto the best available path.

CloudExpress service calculates a value called the Viptela Quality of Experience (vQoE). The vQoE value weighs loss and latency using a formula customized for each application. For example, email applications tolerate latency better than video applications, and video applications tolerate loss better than email applications. The vQoE value ranges from zero to ten, with zero being the worst quality and ten being the best. CloudExpress service computes vQoE values for applications and paths, then assigns applications to the paths that best match their vQoE value. CloudExpress service periodically recalculates vQoE values for paths to ensure ongoing optimal application performance.

TCP Optimization

TCP optimization fine-tunes the processing of TCP data traffic to decrease round-trip latency and improve throughput. You can optimize TCP traffic in service-side VPNs on vEdge routers. Optimizing TCP traffic is especially useful for improving TCP traffic performance on long-latency links, such as transcontinental links and the high-latency transport links used by VSAT satellite communications systems. TCP optimization can also improve the performance of SaaS applications.

TCP optimization is available on vEdge 1000, vEdge 2000, and vEdge 5000 hardware routers.

Additional Information

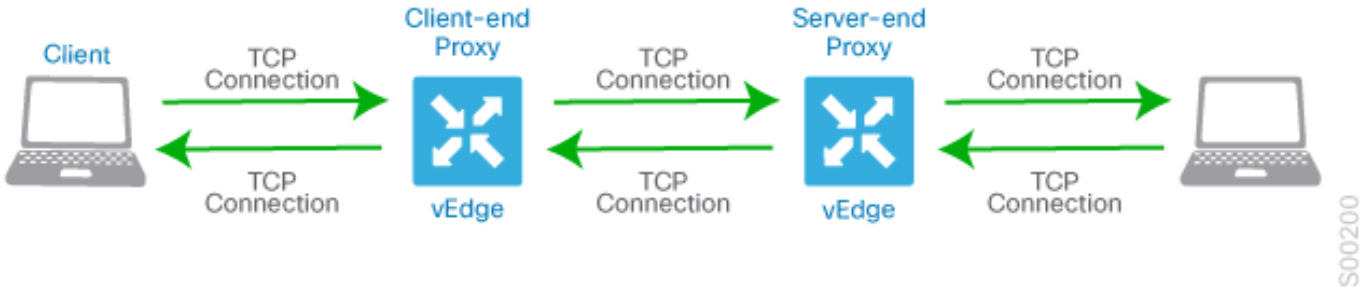
Configuring TCP Optimization
Using CloudExpress Service

Configuring TCP Optimization

TCP optimization fine-tunes the processing of TCP data traffic to decrease round-trip latency and improve throughput. You can optimize TCP traffic in service-side VPNs on vEdge routers. Optimizing TCP traffic is especially useful for improving TCP traffic performance on long-latency links, such as transcontinental links and the high-latency transport links used by VSAT satellite communications systems. TCP optimization can also improve the performance of SaaS applications.

TCP optimization is available on vEdge 1000 and vEdge 2000 hardware routers.

With TCP optimization, a vEdge router acts as a TCP proxy between a client that is initiating a TCP flow and a server that is listening for a TCP flow, as illustrated in the following figure:



This figure shows two vEdge routers acting as proxies. vEdge-1 is the proxy for the client, and is called the client proxy. vEdge-2 is the proxy for the server, called the server proxy. Without TCP optimization, the client establishes a TCP connection directly to the server. When you enable TCP optimization on the two routers, vEdge-1 terminates the TCP connection from client and establishes a TCP connection with vEdge-2. The vEdge-2 router then establishes a TCP connection to the server. The two vEdge routers cache the TCP traffic in their buffers to ensure that the traffic from the client reaches the server without the TCP connection timing out.

It is recommended that you configure TCP optimization on both the vEdge routers, the router closer to the client and the router closer to the server. This configuration is sometimes referred to as a dual-ended proxy. While it is possible to configure TCP optimization only on the vEdge router closer to the client, called single-ended proxy, this configuration is not recommended because the TCP optimization process is compromised. TCP is a bidirectional protocol and operates only when connection-initiation messages (SYNs) are acknowledged by ACK messages in a timely fashion.

If both the client and the server are connected to the same vEdge router, no TCP optimization is performed.

To use TCP optimization, you first enable the feature on the vEdge router. Then you define which TCP traffic to optimize.

Enable TCP Optimization

To use TCP optimization on a vEdge router, you must enable it:

```
vEdge(config-system)# tcp-optimization-enabled
```

On vEdge 1000 and vEdge 2000 routers, enabling TCP optimization carves out a separate CPU core to use for performing TCP optimization, which is a CPU-intensive process.

Optimize TCP in a VPN

To enable TCP optimization for all TCP traffic in a VPN, include the following command when configuring service-side VPNs on a vEdge router:

```
vEdge(config-vpn)# tcp-optimization
```

For example, to enable TCP optimization in VPN 1:

```
vEdge# show running-config vpn 1
vpn 1
interface ge0/4.1
  ip address 10.20.24.15/24
  no shutdown
!
tcp-optimization
!
```

To display information about the TCP flows that the vEdge router is optimizing, use the **show app tcp-opt active-flows** command to view flows that are currently active and the **show app tcp-opt expired-flows** command to view flows that were active but have expired.

Optimize TCP for a Flow

To enable TCP optimization for a specific flow of TCP traffic in a VPN, create a centralized data policy that includes the **tcp-optimization** action. Use the match conditions in the data policy to define the VPN in which to optimize the TCP traffic and the traffic properties to match.

The following example enables TCP optimization on all TCP traffic destined to port 22, which is used for SSH sessions:

```
vSmart# show running-config policy data-policy tcp_optimization_data_policy
policy
 data-policy tcp_optimization_data_policy
  vpn-list vpn_2
   sequence 100
    match
     destination-port 22
    !
    action accept
     count            sample_count
     tcp-optimization
    !
   !
   default-action accept
  !
 !
!

vSmart# show running-config apply-policy
apply-policy site-list tcp_optimization_sites data-policy tcp_optimization_data_policy all
```

Additional Information

Configuring Centralized Data Policy

Configuring Cloud onRamp Service

Cloud onRamp service for IaaS and SaaS extends the fabric of the Viptela overlay network into public cloud instances, allowing branches with vEdge routers to connect directly to public-cloud application providers. By eliminating the need for a physical data center, the Cloud onRamp service improves the performance of SaaS applications.

The connection between the overlay network and a public-cloud application is provided by two redundant vEdge Cloud routers, which act together as a gateway between the overlay network and the application. Using two routers to form the gateway offers path resiliency to the public cloud. In addition, having redundant routers assists in soft failure protection to improve the availability of public-cloud applications. Together, the two routers can remediate link degradation that might occur during soft failures.

Cloud onRamp service discovers any already existing private cloud instances in geographical cloud regions and allows you to select which of them to make available for the overlay network. In such a brownfield scenario, Cloud onRamp service allows simple integration between legacy public-cloud connections and the Viptela overlay network.

You configure and manage Cloud onRamp service through the vManage NMS server. A configuration wizard in the vManage NMS automates the bring-up the gateway to a your public cloud account and automates the connections between public-cloud applications and the users of those applications at branches in the overlay network.

The Cloud onRamp service works in conjunction with AWS virtual private clouds (VPCs).

Cloud onRamp Configuration Overview

To configure Cloud onRamp service, you create gateway VPCs, each of which consists of a pair of vEdge routers. You then map the gateway VPCs to host VPCs that already exist in the AWS cloud.

Gateway VPCs provide the connection between the Viptela overlay network and the cloud-based applications running on host VPCs. Each gateway VPC consists of two vEdge Cloud virtualized routers that reside in their own VPC. Two routers are used to provide redundancy for the connection between the overlay network and cloud-based applications. On each of these two vEdge Cloud routers, the transport VPN (VPN 0) connects to a branch vEdge router, and the service-side VPNs (any VPN except for VPN 0 and VPN 512) connect to applications and application providers in the public cloud.

Host VPCs are virtual private clouds in which your cloud-based applications reside. When a gateway VPC connects to an application or application provider, it is simply connecting to a host VPC.

In the Cloud onRamp configuration process, you map one or more host VPCs to a single gateway VPC. In doing this, you are configuring the cloud-based applications that branch users are able to access.

The mapping process establishes IPsec and BGP connections between the gateway VPC and each host VPC. The IPsec tunnel that connects the gateway VPC and host VPC runs IKE Version 1 to provide security for the connection. The BGP connection that is established over the secure IPsec tunnel allows the gateway VPC and host VPC to exchange routes so that the gateway VPC can direct traffic from the branch to the proper host VPC, and hence to the proper cloud-based application.

During the mapping process, the IPsec tunnels and BGP peering sessions are configured and established automatically. After you establish the mappings, you can view the IPsec and BGP configurations, in the VPN Interface IPsec and BGP feature configuration templates, respectively, and you can modify them as necessary.

Prerequisites for Using Cloud onRamp Service

Before you can configure the Cloud onRamp Service, you must properly provision the vManage NMS and AWS.

vManage NMS Prerequisites

- Make sure that your vManage server has access to the internet and that it has a DNS server configured so that it can reach AWS. To configure a DNS server, in the vManage VPN feature configuration template, enter the IP address of a DNS server, and then reattach the configuration template to the vManage server.
- Ensure that two vEdge Cloud routers that are to be used to bring up the Cloud onRamp service have been added to the vManage NMS and have been attached to the appropriate configuration template. (These two routers are deployed in AWS in their own VPC, and together they form the gateway VPC, which is the bridge between the Viptela overlay network and AWS cloud applications.) Ensure that the configuration for these two routers includes the following:
  - Hostname
  - IP address of vBond orchestrator
  - Site ID
  - Organization name
  - Tunnel interface configuration on the eth1 interface
- Ensure that the vManage NMS is synchronized to the current time. To check the current time, click the Help (?) icon in the top bar of any vManage screen. The Timestamp field shows the current time. If the time is not correct, configure the vManage server's time to point to an NTP time server, such as the Google NTP server. To do this, in the vManage NTP feature configuration template, enter the hostname of an NTP server, and then reattach the configuration template to the vManage server. The Google NTP servers are time.google.com, time2.google.com, time3.google.com, and time4.google.com.

AWS Prerequisites

- Ensure that you have subscribed to the Viptela marketplace Amazon machine images (AMIs) in your AWS account, as described below.
- Ensure that at least one user who has administrative privileges has the AWS API keys for your AWS account. For Cloud onRamp service, these keys are used to authenticate the vManage server with AWS and to bring up the VPC and Elastic Compute Cloud (EC2) instances.
- Check the AWS limits associated with your account (in the Trusted Advisor section of AWS) to ensure that the following resources can be created in your account:
  - 1 VPC, which is required for creating the gateway VPC
  - 6 Elastic IP addresses associated with the gateway vEdge Cloud routers
  - 1 AWS virtual gateway (VGW) for each host VPC
  - 4 VPN connections for mapping each host VPC

Subscribe to Viptela AMIs

To use the Cloud onRamp service and other Viptela services, you must subscribe to the Viptela vEdge router AMI in AWS. In the subscription process, you launch a vEdge Cloud Router AMI instance, you generate a key pair to use for the instance, and finally you use the key pair to subscribe to the vEdge Cloud router instance.

You subscribe to the vEdge Cloud router AMI only once, when you first create a Viptela AMI instance.

To create a new AMI subscription and generate a key pair:

1. In AWS, do a search to locate the Viptela vEdge Cloud Router AMI.
2. Select and launch an EC2 instance with the vEdge AMI instance. For more information, see Create vEdge Cloud VM Instance on AWS.
3. To generate the key pair, in Step 12 in the section Set Up the vEdge Cloud VM Instance , select Create a new key pair.
4. Click Download Key Pair. The key pair is then downloaded to your local computer as a.pem file.
5. Click Launch Instance. A failure message is displayed, because you now need to upload the key pair to complete the subscription process.

To upload the key pair:

1. In AWS Marketplace, http://aws.amazon.com/marketplace , search for the Viptela vEdge Cloud router AMI.
2. Click the Continue button.
3. Click Key Pair to spin up a vEdge Cloud router instance. In the option to enter the key pair, upload the.pem file from your local computer. This is the file that you generated in Step 4, above.

Configure Cloud onRamp Service

To configure Cloud onRamp service, you configure cloud instances using a configuration wizard. Follow the steps below, which are illustrated by this **video** . (Note that the video has no sound.)

1. In vManage NMS, select the Configuration ► Cloud onRamp screen.
2. Click Add New Cloud Instance.
3. In the Add Cloud Instance–Log In to a Cloud Server popup, enter the information to log in to the cloud server:
   a. In the Cloud drop-down, select the cloud type. Currently, this can be only AWS.

b. In the API Key field, enter your Amazon API key.

c. In the Secret Key field, enter the password associated with the API key.

4. Click Log In.
The cloud instance configuration wizard opens. This wizard consists of three screens that you use to configure regions, hosts VPNs, and gateway VPCs, and to map host and gateway VPCs to each other.
A graphic on the right side of each wizard screen illustrates the steps in the cloud instance configuration process. Steps not yet completed are shown in light gray. The current step is highlighted within a blue box. Completed steps are indicated with a green checkmark and are shown in light orange.

5. Select a region and discover host VPCs:

   a. In the Choose Region drop-down, select a geographical region.

   b. Click Discover Host VPCs. A list of host VPCs discovered in that region is displayed.

   c. Select the desired VPCs.

   d. Click Next.

6. Add a gateway VPC:

   a. In the Gateway VPC Name field, type a name for the gateway VPC. The name can be up to 128 characters and can contain only uppercase and lowercase letters, the digits 0 through 9, hyphens (–), and underscores (_). It cannot contain spaces or any other characters.

   b. Under Device Information, enter information about the gateway VPC:

      i. In the vEdge Version drop-down, select the Viptela software version to run on the VPC gateway.

      ii. In the Size of Gateway VPC drop-down, select how much memory and how many CPUs to create on the VPC gateway.

      iii. In the Device 1 drop-down, select the serial number of one of the vEdge Cloud routers that forms the VPC.

      iv. In the Device 2 drop-down, select the serial number of the second vEdge Cloud router.

   c. Click Next.

7. Map the host VPCs to gateway VPCs:

   a. In the table of host VPCs, select the desired host VPCs.

   b. Click Map VPCs. The Map Host VPCs popup opens.

   c. In the Gateway VPC drop-down, select the gateway VPN to map to the host VPCs.

   d. In the VPN drop-down, select the VPN in the overlay network in which to place the mapping.

   e. Click Map VPCs.

   f. Click Save and Complete.

Troubleshoot Cloud onRamp Service

This section describes how to troubleshoot common problems with Cloud onRamp service.

Two vEdge Routers Not Available

Problem Statement

In vManage NMS, when you select the Configuration ► Cloud onRamp screen and click Add New Cloud instance, you see an error message indicating that two vEdge routers are not available.

Resolve the Problem

The vManage NMS does not have two vEdge Cloud routers that are running licensed Viptela software. Contact your operations team so that they can create the necessary vEdge Cloud routers.

If the vEdge routers are present and the error message persists, the two vEdge Cloud routers are not attached to configuration templates. Attach these templates in the vManage Configuration ► Templates ► Device screen. Select the vEdge Cloud router, and then select Attach Devices from the More Actions icon to the right of the row.

Required Permissions for API

Problem Statement

When you enter your API keys, you get an error message indicating that this user does not the the required permissions.

Resolve the Problem

Ensure that the vManage server can reach the internet and has a DNS server configured so that it can reach AWS. To configure a DNS server, in the vManage VPN feature configuration template, enter the IP address of a DNS server, and then reattach the configuration template to the vManage server.

Check the API keys belonging to your AWS account. If you think you have the wrong keys, generate another pair of keys.

If you are entering the correct keys and the error message persists, the keys do not have the required permissions. Check the user permissions associated with the key. Give the user the necessary permissions to create and edit VPCs and EC2 instances.

If the error message persists, check the time of the vManage server to ensure that it is set to the current time. If it is not, configure the vManage server's time to point to the Google NTP server. In the vManage NTP feature configuration template, enter a hostname of time.google.com, time2.google.com, time3.google.com, or time4.google.com. Then reattach the configuration template to the vManage server.

No vEdge Software Versions Appear in the Drop-Down

Problem Statement

When you are trying to configure transit VPC parameters for the gateway VPC, no vEdge Cloud software versions are listed in the drop-down.

Resolve the Problem

Ensure that your customer account has subscribed to the Viptela vEdge Cloud routers.

Also ensure that the vEdge Cloud is running software Release 17.2.0 or later.

No VPNs Appear in Drop-Down

Problem Statement

When you select the host VPCs to map, no VPNs are listed in the drop-down.

Resolve the Problem

This problem occurs when the device configuration template attached to the vEdge Cloud router includes no service-side VPNs. Service-side VPNs (VPNs other than VPN 0 and VPN 512) are required to configure the IPsec connection between the two vEdge Cloud routers selected for the gateway VPC and the host VPC.

This problem can also occur if the two vEdge Cloud routers selected for the gateway VPC have no overlapping service-side VPNs. Because the two vEdges routers form and active–active pair, the same service-side VPNs must be configured on both of them.

To configure service-side VPNs, in the vManage VPN feature configuration template, configure at least one service-side VPN. Ensure that at least one of the service-side VPNs is the same on both routers. Then reattach the configuration template to the routers.

Cloud onRamp Task Fails

Problem Statement

After you have completed mapping the host VPCs to the gateway VPCs, the Cloud onRamp tasks fails.

Resolve the Problem

Review the displayed task information that is displayed on the screen to determine why the task failed. If the errors are related to AWS resources, ensure that all required AWS resources are in place.

Cloud onRamp Task Succeeds, But Routers Are Down

Problem Statement

The Cloud onRamp task was successful, but the vEdge Cloud routers are still in the Down state.

Resolve the Problem

Check the configuration templates:

- Check that all portions of the vEdge Cloud router configuration, including policies, are valid and correct. If the configuration are invalid, they are not applied to the router, so the router never comes up.
- Check that the configuration for the vBond orchestrator is correct. If the DNS name or IP address configured of the vBond orchestrator is wrong, the vEdge router is unable to reach it and hence is unable to join the overlay network.

After you have determined what the configuration issues are:

1. Delete the Cloud onRamp components:
    a. Unmap the host VPNs and the gateway VPCs.
    b. Delete the gateway vEdge routers.
2. Edit the configuration templates and reattach them to the vEdge Cloud routers.
3. Repeat the Cloud onRamp configuration process.

Desired Routes Not Exchanged

Problem Statement

The Cloud onRamp configuration workflow is successful, the Cloud vEdge routers are up and running, but the desired routes are not getting exchanged.

Resolve the Problem

In vManage NMS, check the BGP configuration on the transit vEdge Cloud routers. During the mapping process when you configure Cloud onRamp service, BGP is configured to advertise the network 0.0.0.0/0. Make sure that the service-side VPN contains an IP route that points to 0.0.0.0/0. If necessary, add a static route in the VPN feature configuration templates, and then reattach the configuration to the two vEdge Cloud routers that you selected for the gateway VPC.

On AWS, go to the host VPC and check its route table. In the route table, click the option “Enable route propagation” to ensure that the VPC receives the routes.

End-to-End Ping Is Unsuccessful

Problem Statement

Routing is working properly, but an end-to-end ping is not working.

Resolve the Problem

On AWS, check the security group rules of the host VPC. The security group rules must allow the source IP address range subnets of the on-premises or branch-side devices, to allow traffic from the branch to reach AWS.

Additional Information

Create vEdge Cloud VM Instance on AWS

Using CloudExpress Service

CloudExpress service optimizes the performance of Software as a Service (SaaS) cloud applications based on network loss and latency. CloudExpress service provides clear visibility of the performance of individual applications and automatically chooses the best path for each one.

CloudExpress service calculates a value called the Viptela Quality of Experience (vQoE). The vQoE value weighs loss and latency using a formula customized for each application. For example, email applications tolerate latency better than video applications, and video applications tolerate loss better than email applications. The vQoE value ranges from zero to ten, with zero being the worst quality and ten being the best. CloudExpress service computes vQoE values for applications and paths, and then assigns applications to the paths that best match their vQoE value. CloudExpress service periodically recalculates vQoE values for paths to ensure ongoing optimal application performance.

CloudExpress Requirements

You can enable CloudExpress service in your Viptela overlay network on sites with Direct Internet Access (DIA) and on DIA sites that access the internet through a secure web gateway such as Zscaler or iboss. You can also enable CloudExpress service on client sites that access the internet through another site in the overlay network, called a gateway site. Gateway sites can include regional data centers or carrier-neutral facilities. When you enable CloudExpress service on a client site that accesses the internet through a gateway, you also enable CloudExpress service on the gateway site.

All Viptela devices configured for CloudExpress service must meet these requirements:

- The devices must run Viptela Software Release 16.3 or higher.

- The devices must run in vManage mode.
- You must configure a DNS server address in VPN 0.
- You must configure local exit interfaces in VPN 0:
  - If the local interface list contains only physical interfaces, you must enable NAT on those interfaces. You can use normal default IP routes for next hops.
  - If the local interface list contains only GRE interfaces, you do not need to enable NAT on those interfaces. You can add default routes to the IP address of the GRE tunnel to the destination.

CloudExpress service runs on IPv4 only. It does not support IPv6.

Supported Applications

CloudExpress service supports the following enterprise applications:

- Amazon Web Service (AWS)
- Box
- Concur
- Dropbox
- Google Apps
- GoToMeeting
- Intuit
- Microsoft Office 365
- Oracle
- Salesforce
- SugarCRM
- Zendesk
- Zoho CRM

Enable CloudExpress Service

1. In vManage NMS, select the Administration ► Settings screen.
2. Click the Edit button to the right of the CloudExpress bar.
3. In the Enable CloudExpress field, click Enabled.
4. Click Save.

Configure CloudExpress Service

Add Applications

1. In vManage NMS, select the Configuration ► CloudExpress screen, or click the CloudExpress icon in the title bar. The CloudExpress Dashboard screen opens.
2. From the Manage CloudExpress drop-down, located to the right of the title bar, select Applications to add applications to the CloudExpress configuration.
3. Click the Add Applications and VPN button. The Add Applications & VPN popup window displays.
4. In the Applications field, select an application.
5. In the VPN field, enter the service VPN in which that application runs. You can enter any VPN other than 0 and 512.
6. Click Add.
7. Repeat Steps 3 through 6 for each application you want to add.
8. Click Save Changes.

Configure Client Sites and Gateways

To configure CloudExpress service on client sites that access the internet through gateways, you must configure CloudExpress service both on the client sites and on the gateway sites:

1. In vManage NMS, select the Configuration ► CloudExpress screen, or click the CloudExpress icon in the title bar. The CloudExpress Dashboard screen opens.
2. From the Manage CloudExpress drop-down, located to the right of the title bar, select Client Sites.
3. In the Manage Sites screen, click the Attach Sites button. The Attach Sites screen displays all sites in the overlay network with available sites highlighted. For a site to be available, all devices at that site must be running in vManage mode.
4. In the Available Sites pane, select a client site to attach and click the right arrow. To remove a site, select it in the Selected Sites pane and click the left arrow.
5. Click Attach. vManage NMS pushes the feature template configuration to the devices. The Task View window displays a Validation Success message.
6. Select Configuration ► CloudExpress to return to the CloudExpress Dashboard screen.
7. From the Manage CloudExpress drop-down, located to the right of the title bar, select Gateways.
8. In the Manage Gateways screen, click the Attach Gateways button. The Attach Gateways popup window displays all sites in your overlay network with available sites highlighted. For a site to be available, all devices at that site must be running in vManage mode.
9. In the Available Sites pane, select a gateway site to attach and click the right arrow. To remove a site, select it in the Selected Sites pane and click the left arrow.
10. If you do not specify interfaces for CloudExpress service to use, the system selects a NAT-enabled physical interface from VPN 0. To specify GRE interfaces for CloudExpress service to use:
    a. Click the link Add interfaces to selected sites (optional), located in the bottom right corner of the window.
    b. In the Select Interfaces drop-down, select GRE interfaces to add.
    c. Click Save Changes.
11. Click Attach. vManage NMS pushes the feature template configuration to the devices. The Task View window displays a Validation Success message.

12. To return to the CloudExpress Dashboard, Select Configuration ► CloudExpress.

Configure DIA Sites

1. In vManage NMS, select the Configuration ► CloudExpress screen, or click the CloudExpress icon in the title bar. The CloudExpress Dashboard screen opens.
2. From the Manage CloudExpress drop-down, located to the right of the title bar, select Direct Internet Access (DIA) Sites.
3. In the Manage DIA screen, click Attach DIA Sites. The Attach DIA Sites popup window displays all sites in your overlay network with available sites highlighted. For a site to be available, all devices at that site must be running in vManage mode.
4. In the Available Sites pane, select a site to attach and click the right arrow. To remove a site, select it in the Selected Sites pane and click the left arrow.
5. If you do not specify interfaces for CloudExpress service to use, the system will select a NAT-enabled physical interface from VPN 0. If you would like to specify GRE interfaces for CloudExpress service to use:
    a. Click the link, Add interfaces to selected sites (optional), located in the bottom right corner of the window.
    b. In the Select Interfaces drop-down, select GRE interfaces to add.
    c. Click Save Changes.
6. Click Attach. vManage NMS pushes the feature template configuration to the devices. The Task View window displays a Validation Success message.
7. To return to the CloudExpress Dashboard, Select Configuration ► CloudExpress.

You have now completed configuring the CloudExpress service. To return to the CloudExpress Dashboard, select the Configuration ► CloudExpress screen, or click the CloudExpress icon in the title bar.

View Application Performance

In vManage NMS, select the Configuration ► CloudExpress screen, or click the CloudExpress icon in the title bar. The CloudExpress Dashboard displays the performance of each cloud application in a separate pane.

Each application pane displays the number of vEdge routers accessing the application and the quality of the connection:

- The bottom status bar displays green for devices experience good quality.
- The middle status bar displays yellow for devices experiencing average quality.
- The top status bar displays red for devices experiencing bad quality.

The number to the right of each status bar indicates how many devices are experiencing that quality of connection.

View Application Details

1. In vManage NMS, select the Configuration ► CloudExpress screen, or click the CloudExpress icon in the title bar. The CloudExpress Dashboard displays each cloud application in a separate pane.
2. Click in an application's pane. vManage NMS displays a list of sites accessing the application.
3. Click a graph icon in the vQoE Score column to display vQoE history for that site:

- Click a predefined or custom time period for which to display data.
- Hover over a point on the chart to display vQoE details for that point in time.

Release Information

Introduced in vManage NMS in Release 16.3.

Additional Information

View Application Performance with CloudExpress Service