Overview of Easy Virtual Network

Easy Virtual Network (EVN) is an IP-based virtualization technology that provides end-to-end virtualization of two or more Layer-3 networks. You can use a single IP infrastructure to provide separate virtual networks whose traffic paths remain isolated from each other.

EVN builds on the existing IP-based virtualization mechanism known as VRF-Lite. EVN provides enhancements in path isolation, simplified configuration and management, and improved shared service support. EVN is backward compatible with VRF-Lite to enable seamless network migration from VRF-Lite to EVN.

EVN supports IPv4, static routes, Open Shortest Path First version 2 (OSPFv2), and Enhanced Interior Gateway Routing Protocol (EIGRP) for unicast routing, and Protocol Independent Multicast (PIM) and Multicast Source Discovery Protocol (MSDP) for IPv4 Multicast routing. EVN also supports Cisco Express Forwarding (CEF) and Simple Network Management Protocol (SNMP).

Prerequisites for Configuring EVN

- Implementing EVN in a network requires a single IP infrastructure that you want to virtualize into two or more logical networks or L3VPNs. EVN provides path isolation for the traffic on the different virtual networks.
- You must have a functioning campus design in place before adding virtualization to a network.
- You should understand virtual routing and forwarding (VRF) instances and how they are used to maintain traffic separation across the network.

Restrictions for EVN

- An EVN trunk is allowed on any interface that supports 802.1q encapsulation, such as Fast Ethernet, Gigabit Ethernet, and port channels.

- There are additional platform and line-card restrictions for an EVN trunk. Check Cisco Feature Navigator, www.cisco.com/go/cfn for supported platforms and line cards.
- A single IP infrastructure can be virtualized to provide up to 32 virtual networks end-to-end.
- If an EVN trunk is configured on an interface, you cannot configure VRF-Lite on the same interface.
- OSPFv3 is not supported; OSPFv2 is supported.
- The following are not supported by EVN:
  - IS-IS
  - RIP
  - Route replication is not supported with BGP
  - Certain SNMP set operations
- The following are not supported on an EVN trunk:
  - Access control lists (ACLs)
  - BGP interface commands are not inherited
  - IPv6, except on vnet global
  - Network address translation (NAT)
  - NetFlow
  - Web Cache Communication Protocol (WCCP)

Information About EVN

Benefits of EVN

Easy Virtual Network (EVN) is an IP-based virtualization technology that provides end-to-end virtualization over Layer-3 networks. Network virtualization can be used to secure a network and to reduce network expenses by utilizing the same network infrastructure for multiple virtual networks. You can leverage the same physical infrastructure multiple times by supporting multiple groups, each with their own logical network and unique routing and forwarding tables.

Prior to network virtualization, path isolation can be achieved by:

- Separating pathis using dedicated routers which is more expensive than virtual networks.
- Using access control lists (ACLs), but ACLs do not support unique routing and forwarding tables, can be expensive to maintain, and more prone to error than virtual networks.

EVN provides the following benefits:

- Reduced capital expenditures by not having to maintain separate physical infrastructures to keep traffic isolated. One IP network has two or more virtual networks with traffic path isolation thereby saving the expense of additional hardware.
- Increased business flexibility, due to the ease of network integration for mergers, acquisitions, and business partners.
- Reduced network complexity due to a decrease in the infrastructure requirements for maintaining traffic separation through the core of the network.

- Build on the existing mechanism known as Multi-VRF (VRF-Lite). EVN is compatible with VRF-Lite. See the EVN Compatibility with VRF-Lite section. EVN is recommended over VRF-Lite because EVN provides enhancements in path isolation, simplified configuration and management, and improved shared service support.

  In addition to maintaining traffic separation between business units within a company, there are other scenarios in which path isolation is beneficial, including the following:

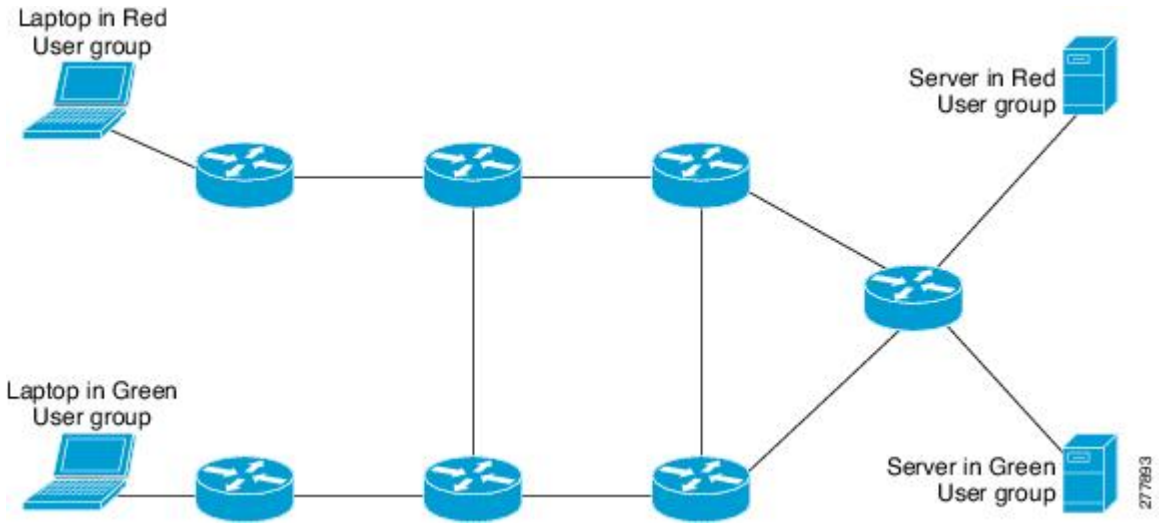
  - Guest access to the Internet—Restricting a guest's network access to the Internet, using a predetermined data path through the customer's network, and being able to define a unique default route for guest traffic.
  - Network Admission Control (NAC) isolation—Isolating the traffic sourced from a noncompliant desktop.
  - Partner access—Restricting partners and contractors to access a network's shared services, such as the Internet, e-mail, DNS, DHCP, or an application server.
  - Application and device isolation—Securing services and devices by "forcing" traffic to a centralized firewall where the traffic is subject to inspection.
  - Outsourcing services—Separating data traffic of various clients from each other.
  - Scalable network—Restricting a portion of the network to traffic that requires a very strict service level, which can lower costs by providing those requirements only where needed.
  - Subsidiaries/mergers/acquisitions—Consolidating companies or networks in stages, while enabling them to share services, when required.
  - Enterprise acting as a service provider—Requiring a separate network under a single authority for autonomous groups. An example is an airport authority supporting a virtual network per airline.

Virtual Network Tags Provide Path Isolation

It is not uncommon to have different user groups running on the same IP infrastructure. Various business reasons require traffic isolation between different groups. The figure below shows two user groups, Red and Green, running on the same network. Prior to network virtualization, there is no separation of traffic between the two groups. Users in the Red user group can access the server in the Green user group, and vice versa.

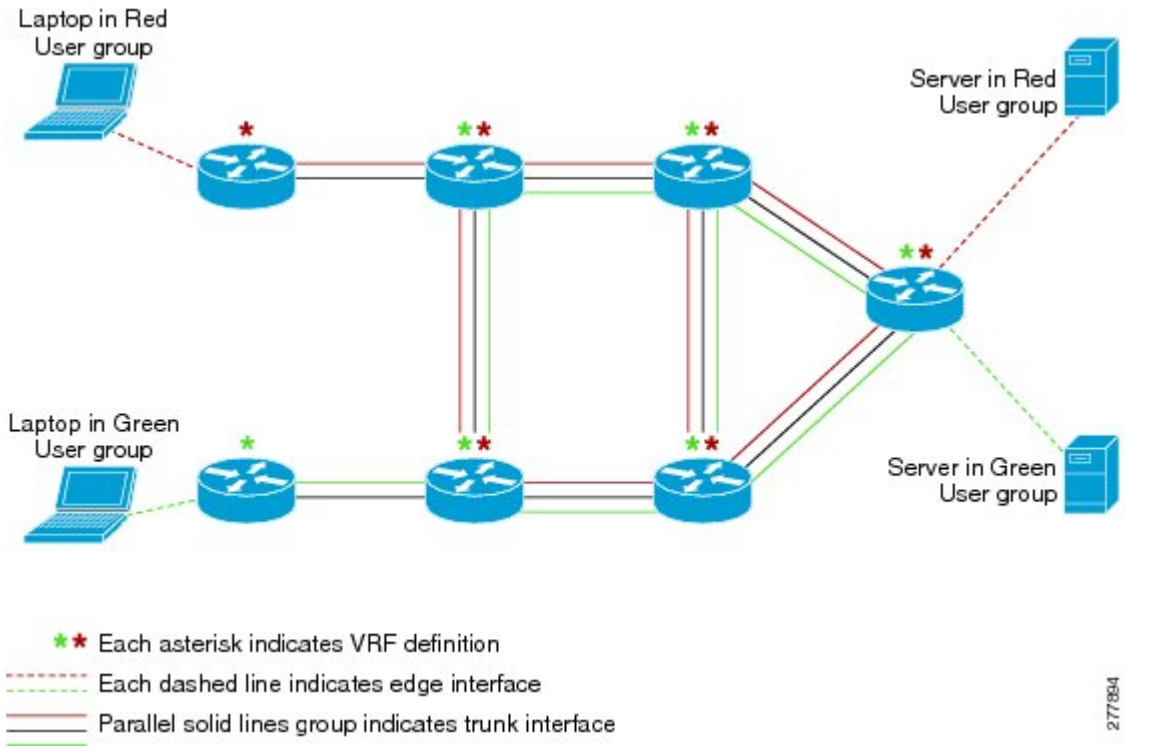
Without network virtualization, path isolation can be achieved by using access control, which is expensive to maintain, prone to error and does not support unique routing and forwarding tables per network.

Figure 1: Network without Virtualization



Virtual networks provide a coarse-grained segmentation of different user groups on one physical network. By configuring virtual networks, you can virtualize a single IP infrastructure to provide a number of virtual networks end to end. In the figure below, a single IP infrastructure is virtualized into two VPNs by creating two VRFs, Red and Green.

Figure 2: Network with Virtualization



In addition to utilizing VRFs to provide device-level separation, each virtual network has path isolation from the other. Path isolation is achieved by tagging the traffic so it carries the same tag value throughout the same

virtual network. Each network device along the path uses the tags to provide separation among different VRFs. A single tag number ties VRF red, for example, on one router to VRF red on another router.

Virtual Network Tag

Each VPN and associated EVN has a tag value that you assign during configuration. The tag value is global, meaning that on each router, the same EVN must be assigned the same numerical tag value. Tag values range from 2 to 4094.

Note When configuring EVN on a Cisco Catalyst 6500 Family networking device, we recommend you assign a vnet tag in the range 2 to1000. Beginning with Cisco IOS Release 15.1(1)SY, on the Sup2T platform of the Cisco Catalyst 6000 product lines, if the **vlan internal allocation policy descending** command is configured, the **vnet tag** range is from 2 to 3900.

An EVN is allowed on any interface that supports 802.1q encapsulation, such as Fast Ethernet, Gigabit Ethernet, and port channels. To allow for backward compatibility with the VRF-Lite solution, the vLAN ID field in the 802.1q frame is used to carry the virtual network tag.

Traffic that carries a virtual network tag is called tagged traffic. Traffic that does not carry a virtual network tag is called untagged traffic.

Tags are illustrated in the following configuration with two VRFs, red and green:

```
! Define two VRFs, red and green.
vrf definition red
 vnet tag 101
!
 address-family ipv4
 exit-address-family
!
vrf definition green
 vnet tag 102
!
 address-family ipv4
 exit-address-family
!
```

A virtual network is defined as a VRF instance with a virtual network tag assigned.

vnet Global

A predefined EVN known as “vnet global” is on the device. It refers to the global routing context and it corresponds to the default RIB. In figure 2 and figure 3, vnet global is represented by a black line connecting routers. The vnet global carries untagged traffic. By default, interfaces belong to the vnet global. Furthermore, vnet global is always running on trunk interfaces. The vnet global is also known as the default routing table.

Note IPv6 traffic is supported in vnet global only.

Edge Interfaces and EVN Trunk Interfaces

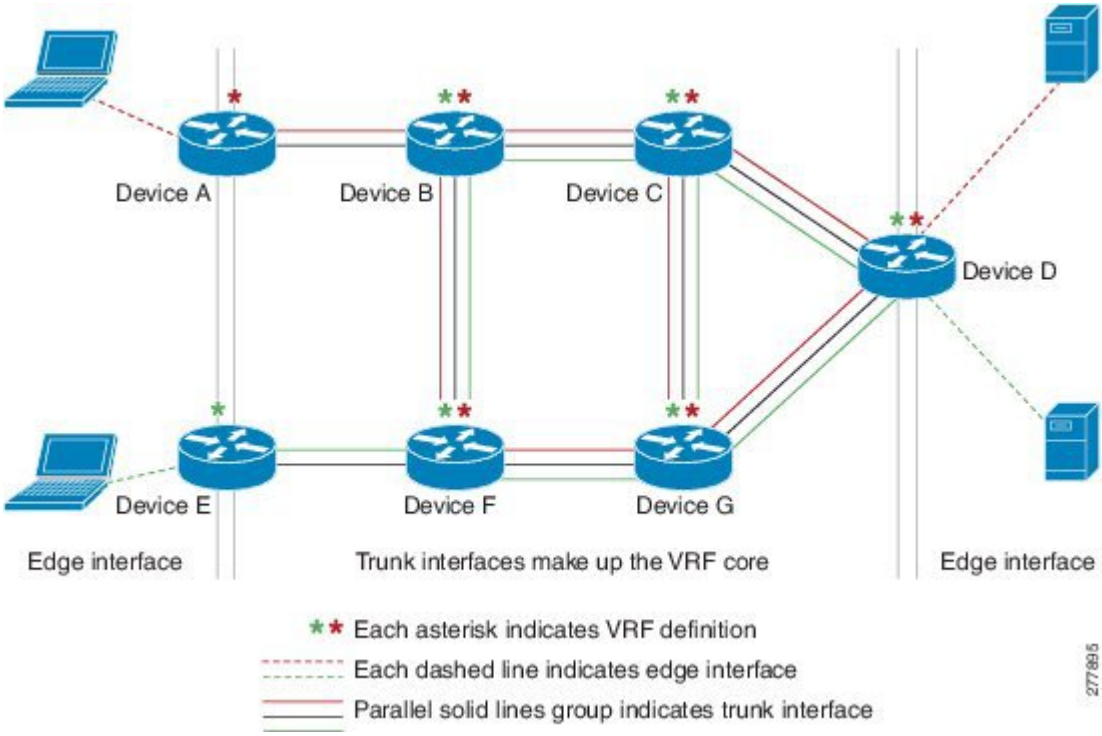
User devices are connected to a Layer 2 switch port, which is assigned to a VLAN. A VLAN can be thought of as a Layer 2 VPN. Customers will group all of the devices that need to be supported in a common Layer 3 VPN in a single VLAN. The point where data traffic is handed off between a VLAN and VRF is called an edge interface.

- An edge interface connects a user device to the EVN and in effect defines the boundary of the EVN. Edge interfaces connect end devices such as hosts and servers that are not VRF-aware. Traffic carried over the edge interface is untagged. The edge interface classifies which EVN the received traffic belongs to. Each edge interface is configured to belong to only one EVN.
- An EVN trunk interface connects VRF-aware routers together and provides the core with a means to transport traffic for multiple EVNs. Trunk interfaces carry tagged traffic. The tag is used to de-multiplex the packet into the corresponding EVN. A trunk interface has one subinterface for each EVN. The **vnet trunk** command is used to define an interface as an EVN trunk interface.

An EVN interface uses two types of interfaces: edge interfaces and trunk interfaces. An interface can be an edge or trunk interface, but not both. Figure 3 illustrates Routers A and D, which have edge interfaces that belong to VRF Red. Routers D and E have edge interfaces that belong to VRF Green.

Routers B, C, D, F, and G have trunk interfaces that make up the EVN core. These five routers have interfaces that belong to both VRF Red and VRF Green.

Figure 3: EVN Edge and EVN Trunk Interfaces



Identifying Trunk Interfaces in Display Output

Because a trunk interface carries multiple EVNs, sometimes it is not sufficient to display only the trunk interface name. When it is necessary to indicate that display output pertains to a particular EVN running on the trunk interface, the convention used is append a period and the virtual network tag, making the format *interface.virtual-network-tag*. Examples are gigabitethernet1/1/1.101 and gigabitethernet1/1/1.102.

By default, when a trunk interface is configured, all of the EVNs and associated virtual network tags are configured, and a virtual network subinterface is automatically created. As stated above, a period and the virtual network tag number are appended to the interface number.

In the following example, VRF red is defined with virtual network tag 3. Hence, the system created Fast Ethernet 0/0/0.3 (in VRF red).

```
Router# show running-config vrf red

Building configuration...
Current configuration : 1072 bytes
vrf definition red
 vnet tag 3
 !
 address-family ipv4
 exit-address-family
 !
```

You can display this hidden interface with the **show derived-config** command and see that all of the commands entered on Fast Ethernet 0/0/0 have been inherited by Fast Ethernet 0/0/0.3:

```
Router# show derived-config interface fastethernet0/0/0.3

Derived configuration : 478 bytes
!
interface FastEthernet0/0/0.3
 description Subinterface for VRF NG red
 vrf forwarding red
 encapsulation dot1Q 3
 ip address 10.1.1.1 255.255.255.0
 ip authentication mode eigrp 1 md5
 ip authentication key-chain eigrp 1 x
 ip bandwidth-percent eigrp 1 3
 ip hello-interval eigrp 1 6
 ip hold-time eigrp 1 18
 no ip next-hop-self eigrp 1
 no ip split-horizon eigrp 1
 ip summary-address eigrp 1 10.0.0.0 255.0.0.0
end
```

Single IP Address on Trunk Interfaces

A trunk interface can carry traffic for multiple EVNs. To simplify the configuration process, all the subinterfaces and associated EVNs have the same IP address assigned. In other words, a trunk interface is identified by the same IP address in different EVN contexts. This is because each EVN has a unique routing and forwarding table, thereby enabling support for overlapping IP addresses across multiple EVNs.

Relationship Between VRFs Defined and VRFs Running on a Trunk Interface

By default, the trunk interfaces on a router will carry traffic for all VRFs defined by the **vrf definition** command. For example, in the following configuration, every VRF defined on the router is included on the interface:

```
interface FastEthernet 1/0/0
 vnet trunk
 ip address 10.1.1.1 255.255.255.0
```

However, you might want to enable only a subset of VRFs over a certain trunk interface for traffic separation purposes. This is achieved by creating a VRF list, which is referenced in the **vnet trunk** command. When a trunk interface is enabled with a VRF list, only VRFs on the list are enabled on the interface. The exception is that **vnet global** is always enabled on the trunk interface.

In the following example, only the two specified VRFs on the list (red and green) are enabled on the interface:

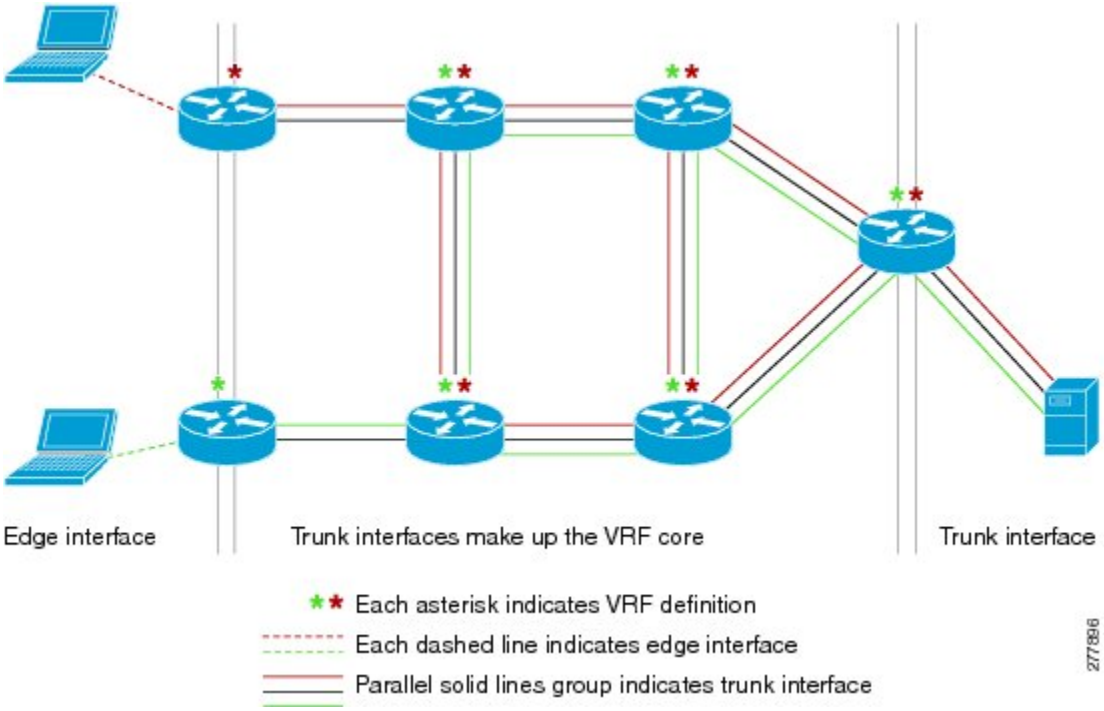
```
vrf list mylist
 member red
 member green
!
interface FastEthernet 1/0/0
 vnet trunk list mylist
 ip address 10.1.1.1 255.255.255.0
```

VRF Awareness

A device connected to a virtual network may not understand virtual network tags and can send and receive only untagged traffic. Such a device is referred to as VRF unaware. For example, a laptop computer is usually VRF unaware.

By contrast, a device that can send and receive tagged traffic and therefore takes the tag value into account when processing such traffic is known as VRF aware. For example, a VRF-aware server shared among different EVNs could use the virtual network tag to distinguish requests received and send responses. A VRF-aware device is connected to the EVN using a trunk interface, as shown in figure 4.

Figure 4: VRF Aware Server



The term “VRF aware” can also be used to describe a software component running on the router. A software component is VRF aware if it can operate on different EVNs. For example, ping is VRF aware because it allows you to choose which EVN to send the ping packet over.

Routing Protocols Supported by EVN

Each EVN runs a separate instance of a routing protocol. This allows each EVN to fine-tune its routing separately and also limits fate sharing. Different virtual networks may run different routing protocols concurrently.

EVN supports static routes, OSPFv2, and EIGRP for unicast routing, and PIM, MSDP, and IGMP for multicast routing.

Packet Flow in a Virtual Network

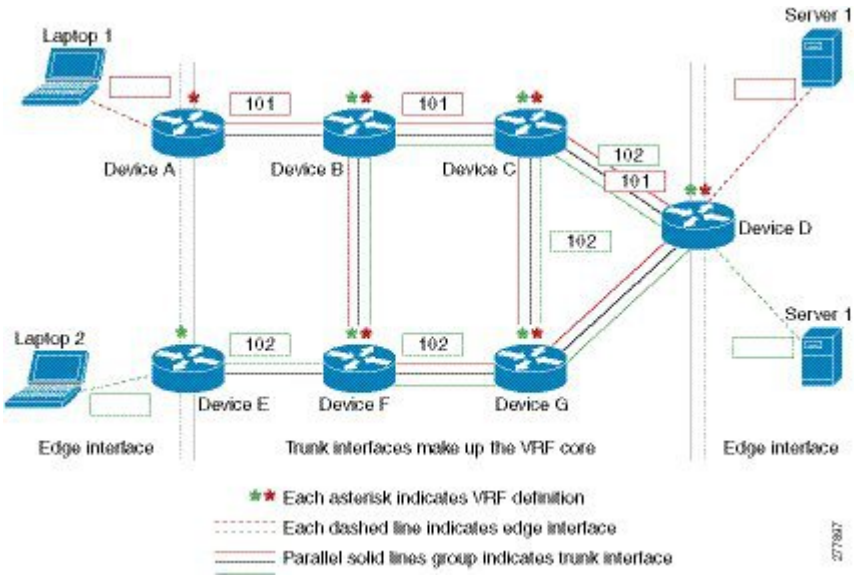
Packets enter an EVN through an edge interface, traverse multiple trunk interfaces, and exit the virtual network through another edge interface. At the ingress edge interface, packets are mapped from a VLAN into a particular EVN. Once the packet is mapped to an EVN, it is tagged with the associated virtual network tag. The virtual network tag allows the trunk interface to carry packets for multiple EVNs. The packets remain tagged until they exit the EVN through the egress edge interface.

On the edge interface, the EVN associated with the interface is used for route lookup. On the trunk interface, the virtual network tag carried in the packet is used to locate the corresponding EVN for routing the packets.

If the egress interface is an edge interface, the packet is forwarded untagged. However, if the egress interface is a trunk interface, the packet is forwarded with the tag of the ingress EVN.

The figure below illustrates how traffic from two VRFs, red and green, can coexist on the same IP infrastructure, using the tags 101 and 102.

Figure 5: Packet Flow in a Virtual Network



The packet flow from Laptop 1 to Server 1 in VRF red occurs as follows:

1. Laptop 1 send an untagged packet to Server 1.
2. Router A receives the packet over an edge interface, which is associated with VRF red.
   a. Router A does route lookup in VRF red and sees that the next hop is Router B through a trunk interface.
   b. Router A encapsulates the packet with VRF red's tag (101) and sends it over the trunk interface.
3. Router B receives the packet over a trunk interface. Seeing virtual network tag 101, Router B identifies that the packet belongs to VRF red.
   a. Router B does route lookup in VRF red and sees that the next hop is Router C through a trunk interface.
   b. Router B encapsulates the packet with VRF red's tag (101) and sends it over the trunk interface.
4. Router C receives the packet over a trunk interface. Using virtual network tag 101, Router C identifies that the packet belongs to VRF red.
   a. Router C does route lookup in VRF red and sees that the next hop is Router D through a trunk interface.
   b. Router C encapsulates the packet with VRF red's tag (101) and sends it over the trunk interface.
5. Router D receives the packet over a trunk interface. Using virtual network tag 101, Router D identifies that the packet belongs to VRF red.
   a. Router D does route lookup in VRF red and sees that the next hop is through an edge interface.
   b. Router D sends the untagged packet over the edge interface to Server 1.
6. Server 1 receives the untagged packet originated from Laptop 1.

Command Inheritance on EVN Trunk Interfaces

One of the benefits of EVN is the ability to easily configure multiple EVNs on a common trunk interface without the need to configure each interface associated with an EVN individually. An EVN trunk interface takes advantage of the fact that the configuration requirements for different EVNs will be similar over a single trunk interface. When specific commands are configured on the trunk interface, they define default values that are inherited by all EVNs running over the same interface, including **vnet global**. If you feel that the settings are acceptable for all of the EVNs sharing an interface, then no individual configuration is necessary.

For example, the OSPF hello interval can be set for all EVNs over the trunk interface with one line of configuration, as follows:

```
interface gigabitethernet1/1/1
 vnet trunk
 ip address 10.1.2.1 255.255.255.0
 ! set OSPF hello interval for all VRFs on this interface.
 ip ospf hello-interval 20
```

The list of commands configured on the trunk interface whose values are inherited by all EVNs running on the same interface is provided in the table in "Commands Whose Values Can be Inherited Or Overridden by a Virtual Network on an Interface" section.

For more examples of command inheritance, see the configuration examples in the *Configuring Easy Virtual Networks* module.

Overriding Command Inheritance Virtual Network Interface Mode

You might want some EVNs on the same trunk interface to have different configurations. An alternative to command inheritance is to selectively override inherited values by using specific commands in virtual network interface mode for individual EVNs. In this mode, the command's settings override the Cisco default value or the value you set in interface configuration mode.

In interface configuration mode, entering the **vnet name** command causes the system to enter virtual network interface mode. The system prompt for this mode is Router(config-if-vnet)#.

The list of commands whose inherited values can be overridden is provided in the table in the "Commands Whose Values Can be Inherited Or Overridden by a Virtual Network on an Interface" section in this module.

Example: Overriding Command Inheritance

In the following example, the OSPF cost of 30 for VRF blue overrides the OSPF cost of 20 for the other VRFs on the interface:

```
interface gigabitethernet 2/0/0
 vnet trunk
 ip address 10.1.1.1 255.255.255.0
 ! Set OSPF cost for all VRFs on this interface to 20.
 ip ospf cost 20
vnet name blue
 description Subinterface for VRF NG blue
 ! Set OSPF cost for blue to 30.
 ip ospf cost 30
```

The **show derived** command indicates the subinterface changed to a cost of 30:

```
Router(config-if-vnet)# do show derived | s interface GigabitEthernet2/0/0
```

```
interface GigabitEthernet2/0/0
vnet trunk
ip address 10.1.1.1 255.255.255.0
ip ospf cost 20
interface GigabitEthernet2/0/0.200
description Subinterface for VRF NG blue
vrf forwarding blue
ip address 10.1.1.1 255.255.255.0
ip ospf cost 30
Router(config-if-vnet)#
```

Example: Enabling an Attribute to vnet Global Only

Similarly, you might want to enable an attribute to vnet global only. To do so, use the **vnet global** interface submode, as follows:

```
interface gigabitethernet1/1/1
 vnet trunk
 ip address 10.1.2.1 255.255.255.0
 vnet global
  ! Set OSPF cost for global to 40.
  ip ospf cost 40
```

In this example, a user wants an EIGRP interface attribute set for all EVNs except vnet global. All EVNs inherit a hold time of 20 seconds, except vnet global, which overrides 20 with a hold time of 40 seconds.

```
interface fastethernet 1/0/0
 vnet trunk
 ip address 10.1.3.1 255.255.255.0
 ip hold-time eigrp 1 20
 vnet global
 ip hold-time eigrp 1 40
```

Removing Overrides and Restoring Values Inherited from EVN Trunk

The **no** and **default** keywords result in different outcomes, depending on whether they are used for a trunk interface or in virtual network interface mode. This section describes the different outcomes.

- When the **no** or **default** keyword is entered before a command on a trunk interface, the trunk is restored to the system's default value for that command. (This is standard behavior resulting for the **no** or **default** keyword).
- When the **default** keyword is entered before a command in virtual network interface mode, the override value is removed and the value that is inherited from the trunk is restored. The override value for the specific EVN is no longer in effect.

In the following example, the trunk interface is configured with an OSPF cost of 20, but VRF blue overrides that value with an OSPF cost of 30:

```
interface gigabitethernet 2/0/0
 vnet trunk
 ip address 10.1.1.1 255.255.255.0
 ! Set OSPF cost for all VRFs on this interface to 20.
 ip ospf cost 20
vnet name blue
 ! Set OSPF cost for blue to 30.
 ip ospf cost 30
```

When the following commands are entered, the OSPF cost value is restored to 20, which is the cost inherited from the trunk interface. (Note that 20 is not the default value of the **ip ospf cost** command.)

```
Router(config-if)# vnet name blue
Router(config-if-vnet)# default ip ospf cost
```

The **default** keyword entered before a command in virtual network interface mode restores the default state, but the **no** keyword does not always do that. In the following example, **no ip dampening-change eigrp 1** disables dampening change.

```
interface Ethernet1/1
 vnet trunk
 ip dampening-change eigrp 1 50
 shutdown
 vnet name red
  no ip dampening-change eigrp 1
! Make sure vnet red does NOT have dampening change enabled, regardless of trunk setting.
!
```

Determining if No Form of Command Appears in Configuration File

If a command is the type of command that switches a feature on or off, the **no** form of the command will appear in the configuration file when configured. That is, nonvolatile generation (NVGEN) overrides the setting from the EVN trunk, as shown in the following example:

```
interface gigabitethernet 2/0/0
 vnet trunk
 ip access-group 1 in
 vnet name red
  no ip pim sparse-mode
  no ip route-cache cef
  no ip access-group in
 vnet global
  ip ospf cost 100
```

If a command takes an argument in its syntax, such as **ip ospf cost** *cost* , the **no** form of the command will remove the configuration, but does not appear in the configuration file. That is, it will not be NVGEN'ed because the user could enter **ip ospf cost** *default-value* to override the inherited value in a more direct way.

EXEC Commands Routing Context

There may be occasions when you want to issue several EXEC commands to apply to a single EVN. In order to reduce the repetitive entering of VRF names for multiple EXEC commands, the **routing-context vrf** command allows you to set the VRF context of EXEC commands once, and then proceed using EXEC commands.

The table below shows four EXEC commands without routing context and in routing context. Note that in the left column, each EXEC command must identify the VRF. In the right column, the VRF content is identified once and the prompt changes to reflect that VRF; there is no need to identify the VRF in each command.

Table 1: EXEC Commands Routing Context

EXEC Commands Without Routing Context	EXEC Commands Routing Context
—	`Router# routing-context vrf red` `Router%red#`
`Router# show ip route vrf red` [Routing table output for VRF red]	`Router%red# show ip route` [Routing table output for VRF red]
`Router# ping vrf red 10.1.1.1` [Ping result using VRF red]	`Router%red# ping 10.1.1.1` [Ping result using VRF red]
`Router# telnet 10.1.1.1 /vrf red` [Telnet to 10.1.1.1 in VRF red]	`Router%red# telnet 10.1.1.1` [Telnet to 10.1.1.1 in VRF red]
`Router# traceroute vrf red 10.1.1.1` [Traceroute output in VRF red]	`Router%red# traceroute 10.1.1.1` [Traceroute output in VRF red]

EVN Compatibility with VRF-Lite

EVN is wire compatible with VRF-Lite. In other words, on the outside, 802.1q, SNMP MIBs, and all the EVN infrastructure will look exactly the same as VRF-Lite.

In the figure below, both routers have VRFs defined. The router on the left uses VRF-Lite, and the router on the right uses an EVN trunk with tags. The two configurations follow the figure.

VRF-Lite Subinterface Configuration

```
interface TenGigabitEthernet1/1/1
 ip address 10.122.5.31 255.255.255.254
 ip pim query-interval 333 msec
 ip pim sparse-mode
 logging event link-status
interface TenGigabitEthernet1/1/1.101
 description Subinterface for Red VRF
 encapsulation dot1Q 101
 ip vrf forwarding Red
 ip address 10.122.5.31 255.255.255.254
 ip pim query-interval 333 msec
 ip pim sparse-mode
```

EVN Trunk Configuration

```
interface TenGigabitEthernet 1/1/1
 vnet trunk
 ip address 10.122.5.32 255.255.255.254
 pim sparse-mode
 logging event link-status
Global Configuration:
vrf definition red
 vnet tag 101

vrf definition green
 vnet tag 102
```

```
 logging event subif-link-status
interface TenGigabitEthernet1/1/1.102
 description Subinterface for Green VRF
 encapsulation dot1Q 102
 ip vrf forwarding Green
 ip address 10.122.5.31 255.255.255.254
 ip pim query-interval 333 msec
 ip pim sparse-mode
 logging event subif-link-status
```

Multiaddress Family VRF Structure

Prior to Cisco IOS Releases 12.2(33)SB and 15.0(1)M, the CLI for a VRF applied to only one address family at a time. For example, the **ip vrf blue** command applies only to the IPv4 address family.

In Cisco IOS Releases 12.2(33)SB and 15.0(1)M, the CLI for a VRF applies to multiple address families under the same VRF. This is known as multiprotocol VRF. For example, the **vrf definition blue** command applies to IPv4 and IPv6 VPNs at the same time, but the routing tables for the two protocols are still different.

Note In Cisco IOS XE Release 3.2S, virtual networks do not support IPv6 except in **vnet global**.

QoS Functionality with EVN

Quality of Service (QoS) configurations are applied to the main physical interface on an EVN trunk. The QoS policy affects all traffic that flows out the physical interface in all the VRFs at the same time. In other words, QoS and network virtualization are mutually independent. For example, traffic marked with the DSCP value specified for voice will be put into the voice queue if the packet is from the red VRF, blue VRF, or green VRF. The traffic for all the VRFs will be queued together.

Commands Whose Values Can be Inherited Or Overridden by a Virtual Network on an Interface

As explained in the "Command Inheritance on EVN Trunk Interfaces" section, there are interface commands that are defined once for a trunk interface, and the value is inherited by each EVN sharing the interface. These commands are sometimes referred to as trunk commands.

A subset of the trunk commands are commands whose values can be overridden by specifying the command in virtual network interface mode. This is explained in the "Overriding Command Inheritance Virtual Network Interface Mode" section.

The table below lists interface commands and indicates whether the values are inherited by the EVNs on the interface and whether the commands can be overridden for a specific EVN.

Table 2: Interface Command Values Inherited or Overridden by a Virtual Network on an Interface

	Values Inherited by EVNs on Interface?	Values Can Be Overriden in Virtual Network Interface Mode?
IP Commands		

	Values Inherited by EVNs on Interface?	Values Can Be Overriden in Virtual Network Interface Mode?
ip accounting	Yes	No
ip address	Yes	No
ip broadcast-address	Yes	No
ip directed broadcast	Yes	No
ip information-reply	Yes	No
ip irdp	Yes	No
ip load-sharing	Yes	No
ip mask-reply	Yes	No
ip mtu	Yes	No
ip proxy-arp	Yes	No
ip redirects	Yes	No
ip unnumbered	Yes	No
ip unreachables	Yes	No
EIGRP Commands		
ip authentication key-chain eigrp	Yes	Yes
ip authentication mode eigrp	Yes	Yes
ip bandwidth-percent eigrp	Yes	Yes
ip dampening-change eigrp	Yes	Yes
ip dampening-interval eigrp	Yes	Yes
ip hello-interval eigrp	Yes	Yes
ip hold-time eigrp	Yes	Yes
ip next-hop-self eigrp	Yes	Yes
ip split-horizon eigrp	Yes	Yes
ip summary-address eigrp	Yes	Yes
Commands that Affect how EIGRP Determines Cost for an Interface		
bandwidth (interface)	Yes	Yes

	Values Inherited by EVNs on Interface?	Values Can Be Overriden in Virtual Network Interface Mode?
delay (interface)	Yes	Yes
OSPF Commands		
ip ospf *process-id* **area**	No	Yes
ip ospf authentication	Yes	Yes
ip ospf authentication-key	Yes	Yes
ip ospf bfd	Yes	Yes
ip ospf cost	Yes	Yes
ip ospf database-filter	Yes	Yes
ip ospf dead-interval	Yes	Yes
ip ospf demand-circuit	Yes	Yes
ip ospf flood-reduction	Yes	Yes
ip ospf hello-interval	Yes	Yes
ip ospf lls	Yes	Yes
ip ospf message-digest-key	Yes	Yes
ip ospf mtu-ignore	Yes	Yes
ip ospf network	Yes	Yes
ip ospf priority	Yes	Yes
ip ospf resync-timeout	Yes	Yes
ip ospf shutdown	Yes	Yes
ip ospf transmit-delay	Yes	Yes
ip ospf transmit-interval	Yes	Yes
ip ospf ttl-security	Yes	Yes
ip ospf vnet area	No	No
IP Multicast Commands		
ip igmp access-group	Yes	Yes
ip igmp explicit-tracking	Yes	Yes

	Values Inherited by EVNs on Interface?	Values Can Be Overriden in Virtual Network Interface Mode?
ip igmp helper-address	Yes	Yes
ip igmp immediate-leave	Yes	Yes
ip igmp last-member-query-count	Yes	Yes
ip igmp last-member-query-interval	Yes	Yes
ip igmp limit	Yes	Yes
ip igmp mroute-proxy	Yes	Yes
ip igmp proxy-service	Yes	Yes
ip igmp querier-timeout	Yes	Yes
ip igmp query-interval	Yes	Yes
ip igmp query-max-response-time	Yes	Yes
ip igmp tcn	Yes	Yes
ip igmp unidirectional-link	Yes	Yes
ip igmp v3lite	Yes	Yes
ip igmp version	Yes	Yes
ip multicast boundary	Yes	Yes
ip pim bidir-neighbor-filter	Yes	Yes
ip pim bsr-border	Yes	Yes
ip pim dense-mode	Yes	Yes
ip pim dr-priority	Yes	Yes
ip pim nbma-mode	Yes	Yes
ip pim neighbor-filter	Yes	Yes
ip pim passive	Yes	Yes
ip pim query-interval	Yes	Yes
ip pim sparse-dense-mode	Yes	Yes
ip pim sparse-mode	Yes	Yes
ip pim state-refresh	Yes	Yes

	Values Inherited by EVNs on Interface?	Values Can Be Overriden in Virtual Network Interface Mode?
Multicast Forwarding Information Base (MFIB) Commands		
ip mfib cef	Yes	Yes
ip mfib forwarding	Yes	Yes

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Easy Virtual Network commands	Easy Virtual Network Command Reference
Configuring Easy Virtual Network	"Configuring Easy Virtual Network" module in the *Easy Virtual Network Configuration Guide*
Configuring Easy Virtual Network shared services and route replication	"Configuring Easy Virtual Network Shared Services" module in the *Easy Virtual Network Configuration Guide*
Easy Virtual Network management and troubleshooting	"Easy Virtual Network Management and Troubleshooting" module in the *Easy Virtual Network Configuration Guide*

MIBs

MIB	MIBs Link
Any MIB that gives VRF information will continue to work with Easy Virtual Network. VRF-independent MIBs report information on every VRF in a system. • CISCO-MVPN-MIB • MPLS-VPN-MIB • CISCO-VRF-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Overview of Easy Virtual Network

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for Overview of Easy VIrtual Network

Feature Name	Releases	Feature Information
EVN VNET Trunk	Cisco IOS XE Release 3.2S 15.0(1)SY 15.1(1)SG Cisco IOS XE Release 3.3SG 15.3(2)T	Easy Virtual Network is an IP-based virtualization technology that provides end-to-end virtualization of the network. You can use a single IP infrastructure to provide separate virtual networks with isolated traffic paths.