Configuring Easy Virtual Network

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

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Prerequisites for Configuring EVN

- Implementing EVN in a network requires a single IP infrastructure that you use to create two or more virtual networks. You want path isolation for traffic on the different virtual networks.
- You should understand the concepts in the “Overview of Easy Virtual Network” module.
- We recommend that you draw your network topology, indicating the interfaces on each router that belong to the EVNs. The diagram facilitates tracking the interfaces you are configuring as edge interfaces and the interfaces you are configuring as trunk interfaces.

How to Configure EVN

Configuring an Easy Virtual Network Trunk Interface

Perform this task to configure an EVN trunk interface, which connects routers to provide the core to transport traffic for multiple virtual networks. Traffic carried over a trunk interface is tagged. This task illustrates how to configure a trunk interface with a base virtual routing and forwarding (VRF) and two named VRFs: VRF red and VRF blue.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf-name*
4. **vnet tag** *number*
5. **description** *string*
6. **address-family ipv4**
7. **exit-address-family**
8. **exit**
9. **vrf definition** *vrf-name*
10. **vnet tag** *number*
11. **description** *string*
12. **address-family ipv4**
13. **exit-address-family**
14. **exit**
15. **interface** *type number*
16. **ip address** *ip-address mask*
17. **vnet trunk** [**list** *vrf-list-name*]
18. **vnet name** *vrf-name*
19. **exit-if-vnet**
20. **no shutdown**
21. **exit**
22. **router ospf** *process-id*
23. **network** *ip-address wildcard* **area** *area-id*
24. **exit**
25. **router ospf** *process-id* **vrf** *vrf-name*
26. **network** *ip-address wildcard* **area** *area-id*
27. **exit**
28. **router ospf** *process-id* **vrf** *vrf-name*
29. **network** *ip-address wildcard* **area** *area-id*
30. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Router(config)# vrf definition red`	Configures a VRF routing table instance and enters VRF configuration mode.
Step 4	**vnet tag** *number* **Example:** `Router(config-vrf)# vnet tag 100`	Specifies the global numeric tag for the VRF. • The same tag number must be configured for the same virtual network on each edge and trunk interface. • When configuring EVN on a Cisco Catalyst 6500 family networking device, we recommend you assign a vnet tag number in the range 2 to 1000.
Step 5	**description** *string* **Example:** `Router(config-vrf)# description guest access`	(Optional) Describes a VRF to help a network administrator review the configuration files.
Step 6	**address-family ipv4** **Example:** `Router(config-vrf)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IP version 4 address prefixes.
Step 7	**exit-address-family** **Example:** `Router(config-vrf-af)# exit-address-family`	Exits address family configuration mode.
Step 8	**exit** **Example:** `Router(config-vrf)# exit`	Exits to global configuration mode.

	Command or Action	Purpose
Step 9	**vrf definition** *vrf-name* **Example:** `Router(config)# vrf definition blue`	Configures a VRF routing table instance and enters VRF configuration mode.
Step 10	**vnet tag** *number* **Example:** `Router(config-vrf)# vnet tag 200`	Specifies the global numeric tag for the VRF. • The same tag number must be configured for the same VRF on each edge and trunk interface.
Step 11	**description** *string* **Example:** `Router(config-vrf) description Finance`	(Optional) Describes a VRF to help a network administrator review configuration files.
Step 12	**address-family ipv4** **Example:** `Router(config-vrf) address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 13	**exit-address-family** **Example:** `Router(config-vrf-af) exit-address-family`	Exits address family configuration mode.
Step 14	**exit** **Example:** `Router(config-vrf)# exit`	Exits to global configuration mode.
Step 15	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 1/1/1`	Configures an interface type and enters interface configuration mode.
Step 16	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.1.1.1 255.255.255.0`	Sets a primary IP address for the interface.
Step 17	**vnet trunk** [**list** *vrf-list-name*]	Defines a trunk interface.

	Command or Action	Purpose
	Example: `Router(config-if)# vnet trunk`	• By default, all VRFs defined with the **vrf definition** command run on all trunk interfaces on the router. Therefore, VRF red and VRF blue are now running on this interface. • Use the **list** *vrf-list-name* command elements to restrict VRFs running on a trunk interface.
Step 18	**vnet name** *vrf-name* **Example:** `Router(config-if)# vnet name red`	(Optional) Enters virtual network interface mode to configure features that apply to a specified VRF to override global VRF values. • This step is not necessary if the global settings are acceptable for all of the VRFs on the interface. • After this step, you configure one or more eligible commands, such as **ip ospf cost**. (Not shown in this task.) For the list of commands that are used to override global VRF values, see Overview of Easy Virtual Network module, Table 2.
Step 19	**exit-if-vnet** **Example:** `Router(config-if-vnet) exit-if-vnet`	Exits VRF interface configuration mode and enters interface configuration mode.
Step 20	**no shutdown** **Example:** `Router(config-if) no shutdown`	Restarts an interface.
Step 21	**exit** **Example:** `Router(config-if) exit`	Exits to global configuration mode.
Step 22	**router ospf** *process-id* **Example:** `Router(config)# router ospf 1`	Configures an Open Shortest Path First (OSPF) routing process and associates it with a VRF. • This OSPF instance has no VRF, so it is **vnet global**.
Step 23	**network** *ip-address wildcard* **area** *area-id* **Example:** `Router(config-router) network 10.0.0.0 255.255.255.0 area 0`	Defines the interfaces and associated area IDs on which OSPF runs.

	Command or Action	Purpose
Step 24	**exit** **Example:** `Router(config-router) exit`	Exits to global configuration mode.
Step 25	**router ospf** *process-id* **vrf** *vrf-name* **Example:** `Router(config)# router ospf 2 vrf red`	Configures an OSPF routing process and associates it with a VRF. • Specifies a different *process-id* for each VRF because they each need their own OSPF instance.
Step 26	**network** *ip-address wildcard* **area** *area-id* **Example:** `Router(config-router) network 10.0.0.0 255.255.255.0 area 0`	Defines the interfaces and associated area IDs on which OSPF runs and the area ID for those interfaces.
Step 27	**exit** **Example:** `Router(config-router) exit`	Exits to global configuration mode.
Step 28	**router ospf** *process-id* **vrf** *vrf-name* **Example:** `Router(config)# router ospf 3 vrf blue`	Configures an OSPF routing process and associates it with a VRF. • Specifies a different *process-id* for each VRF because they each need their own OSPF instance.
Step 29	**network** *ip-address wildcard* **area** *area-id* **Example:** `Router(config-router) network 10.0.0.0 255.255.255.0 area 2`	Defines the interfaces and associated area IDs on which OSPF runs and the area ID for those interfaces.
Step 30	**end** **Example:** `Router(config-vrf) end`	Ends the configuration session and returns to privileged EXEC mode.

Enabling a Subset of VRFs over a Trunk Interface

The prior task, “Configuring an Easy Virtual Network Trunk Interface,” shows how to configure a trunk interface with two VRFs. By default, the trunk interfaces on a router can carry traffic for each VRF defined by the **vrf definition** command. However, you might want to enable only a subset of VRFs over a trunk

interface, which is done by creating a VRF list. A maximum of 32 VRF lists can exist on a router. Perform the following task to create a VRF list. This task presumes that the VRF has already been configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf list** *vrf-list-name*
4. **member** *vrf-name*
5. Repeat Step 4 to add other VRFs to the list.
6. **exit-vrf-list**
7. **interface** *type number*
8. **vnet trunk list** *vrf-list-name*
9. **ip address** *ip-address mask*
10. **end**
11. **show vrf list** [*vrf-list-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vrf list** *vrf-list-name* **Example:** `Router(config)# vrf list External`	Defines a list of VRFs and enters VRF list configuration mode. • The *vrf-list-name* argument may contain up to 32 characters. Quotation marks, spaces, and * are not allowed.
Step 4	**member** *vrf-name* **Example:** `Router(config-vrf-list)# member blue`	Specifies an existing VRF as a member of a VRF list. • The VRF must be defined before it can be added to a list.
Step 5	Repeat Step 4 to add other VRFs to the list.	(Optional) If you want a trunk interface with one VRF, your list only needs one VRF.

	Command or Action	Purpose
Step 6	**exit-vrf-list** **Example:** `Router(config-vrf-list)# exit-vrf-list`	Exits VRF list configuration mode.
Step 7	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 1/1/1`	Configures an interface and enters interface configuration mode.
Step 8	**vnet trunk list** *vrf-list-name* **Example:** `Router(config-if)# vnet trunk list mylist`	Defines a trunk interface and enables the VRFs that are in the VRF list. • Use the *vrf-list-name* defined in Step 3.
Step 9	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.1.3.1 255.255.255.0`	Sets a primary IP address for the interface.
Step 10	**end** **Example:** `Router(config-if) end`	Ends the configuration session and returns to privileged EXEC mode.
Step 11	**show vrf list** [*vrf-list-name*] **Example:** `Router# show vrf list mylist`	Displays information about a VRF list.

Configuring an EVN Edge Interface

Perform this task to configure an edge interface, which connects a user device to a virtual network. Traffic carried over an edge interface is untagged. The edge interface determines which virtual network the received traffic belongs to. Each edge interface is mapped to only one virtual network.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **vrf forwarding** *vrf-name*
5. **ip address** *ip-address mask*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 1/0/0`	Configures an interface type and enters interface configuration mode.
Step 4	**vrf forwarding** *vrf-name* **Example:** `Router(config-if)# vrf forwarding red`	Defines an edge interface and determines the VRF that the incoming traffic belongs to. • The *vrf-name* must already be defined by a **vrf definition** command. • In this example, incoming traffic belongs to VRF red. **Note** Make sure you are not on the trunk interface when you are trying to configure an edge interface.
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.1.1.1 255.255.255.0`	Sets a primary IP address for the interface.

	Command or Action	Purpose
Step 6	**end** **Example:** `Router(config-if) end`	Ends the configuration session and returns to privileged EXEC mode.

What to Do Next

After you have configured an edge interface and a trunk interface, refer to your network diagram and log on to a different router. If it is has an edge interface, configure that interface. If it has a trunk interface, configure that interface with the appropriate VRFs. Continue configuring each of the routers and interfaces that belong to each VRF.

Configure other protocol features you want running in your VRFs. See the appropriate IP Routing configuration guide.

Verifying EVN Configurations

Perform any of the following steps in this task to verify your configuration. Because a virtual network is a VRF, all the existing VRF **show** commands are supported for virtual networks. If a router has a mix of VRFs and virtual networks, the various **show vrf** commands will include both VRFs and virtual networks in the output.

SUMMARY STEPS

1. **enable**
2. **show vnet tag**
3. **show running-config** [**vrf** | **vnet**] [*vrf-name*]
4. **show vrf list** [*vrf-list-name*]
5. **show** {**vrf** | **vnet**} [**ipv4** | **ipv6**] [**interface** | **brief** | **detail** | **lock**] [*vrf-name*]
6. **show** {**vrf** | **vnet**} **counters**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show vnet tag** **Example:** `Router# show vnet tag`	(Optional) Displays where each tag has been configured or used.
Step 3	**show running-config** [**vrf** | **vnet**] [*vrf-name*] **Example:** `Router# show running-config vrf green`	(Optional) Displays the VRFs in the running configuration, displays the interfaces in the VRFs, and displays the protocol configurations for Multi-VRF.
Step 4	**show vrf list** [*vrf-list-name*] **Example:** `Router# show vrf list`	(Optional) Displays information about VRF lists, such as the VRFs in each list.
Step 5	**show** {**vrf** | **vnet**} [**ipv4** | **ipv6**] [**interface** | **brief** | **detail** | **lock**] [*vrf-name*] **Example:** `Router# show vnet detail`	(Optional) Displays information about the VRFs.
Step 6	**show** {**vrf** | **vnet**} **counters** **Example:** `Router# show vnet counters`	(Optional) Displays information about the number of VRFs or virtual networks supported and configured.

Configuration Examples for Configuring EVN

Example: Virtual Networks Using OSPF with network Commands

In this example, **network** commands associate a shared VRF interface with a base VRF and two named VRFs, red and blue. There are three OSPF instances because each VRF needs its own OSPF instance. OSPF 1 has no VRF, so it is **vnet global**.

```
vrf definition red
 vnet tag 100
 address-family ipv4
 exit-address-family
!
vrf definition blue
 vnet tag 200
 address-family ipv4
 exit-address-family
```

```
!
interface gigabitethernet 0/0/0
 ip address 10.0.0.1 255.255.255.0
 vnet trunk
 vnet name red
  ip ospf cost 100
!
router ospf 1
 log-adjacency-changes detail
 network 10.0.0.0 255.255.255.0 area 0
router ospf 2 vrf red
 log-adjacency-changes
 network 10.0.0.0 255.255.255.0 area 0
router ospf 3 vrf blue
 log-adjacency-changes
 network 10.0.0.0 255.255.255.0 area 2
```

Example: Virtual Networks Using OSPF with ip ospf vnet area Command

This example differs from the prior example regarding the association between OSPF instances and a particular interface. In this example, OSPF is running on all of the virtual networks of a trunk interface. The **ip ospf vnet area** command associates the GigabitEthernet 0/0/0 interface with the three OSPF instances.

```
vrf definition red
 vnet tag 100
 address-family ipv4
 exit-address-family
!
vrf definition blue
 vnet tag 200
 address-family ipv4
 exit-address-family
!
interface gigabitethernet 0/0/0
 ip address 10.0.0.1 255.255.255.0
 vnet trunk
 ip ospf vnet area 0
 vnet name red
  ip ospf cost 100
 vnet name blue
  ip ospf 3 area 2
!
router ospf 1
 log-adjacency-changes detail
router ospf 2 vrf red
 log-adjacency-changes
router ospf 3 vrf blue
 log-adjacency-changes
```

Example: Command Inheritance and Virtual Network Interface Mode Override in an EIGRP Environment

This example shows a GigabitEthernet interface configured with various EIGRP commands:

```
interface gigabitethernet0/0/0
 vnet trunk
 ip address 10.0.0.1 255.255.255.0
 ip authentication mode eigrp 1 md5
 ip authentication key-chain eigrp 1 x
 ip bandwidth-percent eigrp 1 3
 ip dampening-change eigrp 1 30
 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 1.0.0.0 255.0.0.0
end
```

Because a trunk is configured, a VRF subinterface is automatically created and the commands on the main interface are inherited by the VRF subinterface (g0/0/0.3, where the number 3 is the tag number from vnet tag 3 .)

```
R1# show running-config vrf red
Building configuration...
Current configuration : 1072 bytes
vrf definition red
 vnet tag 3
 !
 address-family ipv4
 exit-address-family
 !
```

If you display that hidden subinterface with the **show derived-config** command, you'll see that all of the commands entered on GigabitEthernet 0/0/0 have been inherited by GigabitEthernet 0/0/0.3:

```
R1# show derived-config interface gigabitethernet0/0/0.3
Building configuration...
Derived configuration : 478 bytes
!
interface GigabitEthernet0/0/0.3
 description Subinterface for VNET red
 vrf forwarding red
 encapsulation dot1Q 3
 ip address 10.0.0.1 255.255.255.0
 ip authentication mode eigrp 1 md5
 ip authentication key-chain eigrp 1 x
 ip bandwidth-percent eigrp 1 3
 ip dampening-change eigrp 1 30
 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 1.0.0.0 255.0.0.0
end
```

You can override those commands by using virtual network interface mode (under the **vnet name** command). For example:

```
R1(config)# interface gigabitethernet0/0/0
R1(config-if)# vnet name red
R1(config-if-vnet)# no ip authentication mode eigrp 1 md5
 ! disable authen for e0/0.3 only
R1(config-if-vnet)# ip authentication key-chain eigrp 1 y
 ! different key-chain
R1(config-if-vnet)# ip band eigrp 1 99
 ! higher bandwidth-percent
R1(config-if-vnet)# no ip dampening-change eigrp 1
 ! disable dampening-change
R1(config-if-vnet)# ip hello eigrp 1 7
R1(config-if-vnet)# ip hold eigrp 1 21
R1(config-if-vnet)# ip next-hop-self eigrp 1
 ! enable next-hop-self for e0/0.3
R1(config-if-vnet)# ip split-horizon eigrp 1
 ! enable split-horizon
R1(config-if-vnet)# no ip summary-address eigrp 1 10.0.0.1 255.0.0.0
 ! do not summarize on e0/0.3


R1(config-if-vnet)# do show running-config interface gigabitethernet0/0/0

Building configuration...
Current configuration : 731 bytes
```

```
!
interface GigabitEthernet0/0/0
 vnet trunk
 ip address 1.1.1.1 255.255.255.0
 ip authentication mode eigrp 1 md5
 ip authentication key-chain eigrp 1 x
 ip bandwidth-percent eigrp 1 3
 ip dampening-change eigrp 1 30
 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 1.0.0.0 255.0.0.0
 vnet name red
  ip split-horizon eigrp 1
  no ip summary-address eigrp 1 1.0.0.0 255.0.0.0
  no ip authentication mode eigrp 1 md5
  ip authentication key-chain eigrp 1 y
  ip bandwidth-percent eigrp 1 99
  no ip dampening-change eigrp 1
  ip hello-interval eigrp 1 7
  ip hold-time eigrp 1 21
  ip next-hop-self eigrp 1
 !
end
```

Notice that g0/0.3 is now using the override settings:

```
R1(config-if-vnet)# do show derived-config interface g0/0.3

Building configuration...
Derived configuration : 479 bytes
!
interface GigabitEthernet0/0/0.3
 description Subinterface for VNET red
 vrf forwarding red
 encapsulation dot1Q 3
 ip address 1.1.1.1 255.255.255.0
 no ip authentication mode eigrp 1 md5
 ip authentication key-chain eigrp 1 y
 ip bandwidth-percent eigrp 1 99
 no ip dampening-change eigrp 1
 ip hello-interval eigrp 1 7
 ip hold-time eigrp 1 21
 ip next-hop-self eigrp 1
 ip split-horizon eigrp 1
 no ip summary-address eigrp 1 1.0.0.0 255.0.0.0
end
```

Commands entered in **vnet name** submode are sticky. That is, when you enter a command in **vnet name** submode, it will nvgen, regardless of whether it is set to the same value as the default value. For example, the default hello value is 5. When the **ip hello eigrp** command is entered in **vnet name** submode, it will nvgen; it does not do that in any other mode.

```
R1(config-if)# interface gigabitethernet0/0/2
R1(config-if)# vnet trunk
R1(config-if)# ip bandwidth-percent eigrp 1 50         <---<< this will NOT nvgen
R1(config-if)# ip hello eigrp 1 5              <---<< this will NOT nvgen
R1(config-if)# no ip authentication mode eigrp 1 md5    <---<< this will NOT nvgen
R1(config-if)# vnet name red
R1(config-if-vnet)# ip bandwidth-percent eigrp 1 50   <---<< this will nvgen
R1(config-if-vnet)# ip hello eigrp 1 5   <---<< this will nvgen
R1(config-if-vnet)# no ip authentication mode eigrp 1 md5  <---<< this will nvgen
R1(config-if-vnet)# do show running-config interface gigabitethernet0/0/2

Building configuration...
Current configuration : 104 bytes
!
interface GigabitEthernet0/0/2
 vnet trunk
```

```
 no ip address
 vnet name red
  ip bandwidth-percent eigrp 1 50
  ip hello-interval eigrp 1 5
  no ip authentication mode eigrp 1 md5
 !
```

Because of this sticky factor, to remove a configuration entry in **vnet name** submode, you typically must use the default form of that command. Some commands can also be removed using the **no** form; it depends on the command. Some commands use the **no** form to disable the command instead, such as the **authentication** and **summary-address** commands.

```
R1(config-if-vnet)# default ip authentication mode eigrp 1 md5
R1(config-if-vnet)# no ip bandwidth-percent eigrp 1
R1(config-if-vnet)# no ip hello eigrp 1

R1(config-if-vnet)# do show running-config interface g0/2

Building configuration...
Current configuration : 138 bytes
!
interface GigabitEthernet0/0/2
 vnet trunk
 no ip address
 vnet name red
 !
end
```

Example: Command Inheritance and Virtual Network Interface Mode Override in a Multicast Environment

The following example illustrates command inheritance and virtual network interface mode override in a multicast network. A trunk interface leverages the fact that configuration requirements from different VRFs will be similar over the same trunk interface. Eligible commands configured on the trunk interface are inherited by all VRFs running over the same interface.

In this example, IP multicast (PIM sparse mode) is configured on the trunk interface, which has several VRFs:

```
vrf definition red
 vnet tag 13
 !
 address-family ipv4
 exit-address-family
!
ip multicast-routing
ip multicast-routing vrf red
interface GigabitEthernet0/1/0
 vnet trunk
 ip address 125.1.15.18 255.255.255.0
 ip pim sparse-mode
```

The user decides that he does not want IP multicast configured for VRF red on GigabitEthernet 0/1/0, so he uses the virtual network interface mode override. IP Multicast is disabled for VRF red only. The **no ip pim** command disables all modes of Protocol Independent Multicast (PIM), including sparse mode, dense mode, and sparse-dense mode, for VRF red.

```
interface GigabitEthernet0/1/0
 vnet trunk
 ip address 125.1.15.18 255.255.255.0
 ip pim sparse-mode
 vnet name red
 no ip pim
```

Example: EVN Using IP Multicast

The following example configures PIM sparse mode and leverages Anycast RP for RP redundancy. In this example, only one VRF is configured.

The example shows how to enable multicast routing globally and on each L3 interface. The black text indicates the group of commands configuring the global table; the red text indicates the group of commands configuring VRF red.

```
ip multicast-routing
interface GigabitEthernet 1/1/1
 description GigabitEthernet to core (Global)                    GLOBAL TABLE
 ip pim sparse-mode
vrf definition red
 vnet tag 100
!
 address-family ipv4
 exit-address-family
!
ip multicast-routing vrf red                                      VRF RED
!
interface gigabitethernet1/1/1.100
 description GigabitEthernet to core (VRF red)
 vrf forwarding red
 ip pim sparse-mode
```

Configure the RP in the VRF using Anycast RP.

```
interface loopback0
 description Anycast RP Global
 ip address 10.122.5.200 255.255.255.255
 ip pim sparse-mode
!
interface loopback1
 description MDSP Peering interface
 ip address 10.122.5.250 255.255.255.255                     GLOBAL TABLE
 ip pim sparse-mode
!
ip msdp peer 10.122.5.251 connect-source loopback 1
ip msdp originator-id loopback 1
ip pim rp-address 10.122.5.200
access-list 10 permit 239.0.0.0 0.255.255.255
!
!
interface loopback 10
 description Anycast RP VRF Red
 vrf forwarding red
 ip address 10.122.15.200 255.255.255.255
 ip pim sparse-mode
interface loopback 11
 description MSDP Peering interface VRF red                    VRF RED
 vrf forwarding red
 ip address 10.122.15.250 255.255.255.255
 ip pim sparse-mode
!
ip msdp vrf red peer 10.122.15.251 connect-source loopback 11
ip msdp vrf red originator-id loopback 11
!
ip pim vrf red rp-address 10.122.15.200
access-list 11 permit 239.192.0.0 0.0.255.255
```

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
Information about Easy Virtual Network configuration tasks	"Overview of Easy Virtual Networks" module in the *Easy Virtual Network Configuration Guide*
Easy Virtual Network shared services and route replication configuration tasks	"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 EVN. 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 Configuring 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 1: Feature Information for Configuring 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	This module describes how to configure virtual IP networks. An EVN 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 whose traffic paths remain isolated from each other. The following commands were modified: **vrf definition** , **vrf forwarding** . The following commands were introduced: **description** (vrf definition submode), **exit-if-vnet** , **exit-vrf-list** , **member** (vrf list), **routing-context** , **show running-config vnet** , **show vnet** , **show vnet counters** , **show vnet tag** , **show vrf counters** , **show vrf list** , **vnet** , **vnet tag** , **vnet trunk** , **vrf list** .
EVN OSPF	Cisco IOS XE Release 3.2S 15.0(1)SY 15.1(1)SG Cisco IOS XE Release 3.3SG 15.3(2)T	EVN OSPF provides Easy Virtual Network support for OSPF. The following commands were modified: **ip ospf database-filter all out** , **ip ospf demand-circuit** , **ip ospf flood-reduction** , **ip ospf mtu-ignore** , **ip ospf shutdown** . The following command was introduced: **ip ospf vnet area** .

Feature Name	Releases	Feature Information
EVN EIGRP	Cisco IOS XE Release 3.2S 15.0(1)SY 15.1(1)SG Cisco IOS XE Release 3.3SG 15.3(2)T	EVN EIGRP provides Easy Virtual Network support for EIGRP. The following commands were modified: **ip summary-address eigrp** , **summary-metric** .
EVN Multicast	Cisco IOS XE Release 3.2S 15.0(1)SY 15.1(1)SG Cisco IOS XE Release 3.3SG 15.3(2)T	EVN Multicast provides Easy Virtual Network support for IP Multicast.