PBR Next-Hop Verify Availability for VRF

The PBR Next-Hop Verify Availability for VRF feature enables verification of next-hop availability for IPv4/IPv6 packets in virtual routing and forwarding (VRF) instances.

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.

Information About PBR Next-Hop Verify Availability for VRF

PBR Next-Hop Verify Availability for VRF Overview

Cisco IOS policy-based routing (PBR) defines packet matching and classification specifications, sets action policies, which can modify the attributes of IP packets, and overrides normal destination IP address-based routing and forwarding. PBR can be applied on global interfaces and under multiple routing instances. The PBR Next-Hop Verify Availability for VRF feature enables verification of next-hop availability for IPv4/IPv6 packets under virtual routing and forwarding (VRF) instances.

In case of an inherited VRF, the VRF instance is based on the ingress interface. Inter VRF refers to forwarding of packets from one VRF to another VRF; for example, from VRFx to VRFy. An IPv4/IPv6 packet received from VRFx is forwarded to VRFy and the availability of the next hop is verified in the VRFy instance.

How to Configure PBR Next-Hop Verify Availability for VRF

Configuring PBR Next-Hop Verify Availability for Inherited IP VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *vpn-route-distinguisher*
5. **route-target export** *route-target-ext-community*
6. **route-target import** *route-target-ext-community*
7. **exit**
8. **ip sla** *operation-number*
9. **icmp-echo** *destination-ip-address*
10. **vrf** *vrf-name*
11. **exit**
12. **ip sla schedule** *operation-number* **life forever start-time now**
13. **track** *object-number* **ip sla** *operation-number*
14. **interface** *type number*
15. **ip vrf forwarding** *vrf-name*
16. **ip address** *ip-address subnet-mask*
17. **exit**
18. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] [
19. **set ip vrf** *vrf-name* **next-hop verify-availability** *next-hop-address sequence* **track** *object*
20. **exit**
21. **interface** *type number*
22. **ip vrf forwarding** *vrf-name*
23. **ip policy route-map** *map-tag*
24. **ip address** *ip-address subnet-mask*
25. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf RED`	Configures an IP VPN routing and forwarding instance and enters VRF configuration mode.
Step 4	**rd** *vpn-route-distinguisher* **Example:** `Device(config-vrf)# rd 100:1`	Specifies the route distinguisher. The route distinguisher is either an autonomous system (AS) number or an IP address.
Step 5	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 100:1`	Creates a route-target extended community for a VRF and exports routing information from the target VPN extended community. The *route-target-ext-community* argument is either an AS number or an IP address.
Step 6	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 100:1`	Creates a route-target extended community for a VRF and imports routing information from the target VPN extended community. The *route-target-ext-community* argument is either an AS number or an IP address.
Step 7	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and returns to global configuration mode.
Step 8	**ip sla** *operation-number* **Example:** `Device(config)# ip sla 1`	Configures a Cisco IOS IP Service Level Agreements (SLAs) operation and enters IP SLA configuration mode.
Step 9	**icmp-echo** *destination-ip-address* **Example:** `Device(config-ip-sla)# icmp-echo 10.0.0.4`	Configures an IP SLAs Internet Control Message Protocol (ICMP) echo operation and enters ICMP echo configuration mode.
Step 10	**vrf** *vrf-name* **Example:** `Device(config-ip-sla-echo)# vrf RED`	Configures IP SLAs for a VRF instance.
Step 11	**exit** **Example:** `Device(config-ip-sla-echo)# exit`	Exits ICMP echo configuration mode and returns to global configuration mode.
Step 12	**ip sla schedule** *operation-number* **life forever start-time now** **Example:** `Device(config)# ip sla schedule 1 life forever start-time now`	Configures the scheduling parameters for a single Cisco IOS IP SLAs operation.
Step 13	**track** *object-number* **ip sla** *operation-number* **Example:**	Tracks the state of a Cisco IOS IP SLAs operation and enters tracking configuration mode.

	Command or Action	Purpose
	`Device(config)# track 1 ip sla 1`	
Step 14	**interface** *type number* **Example:** `Device(config-track)# interface Ethernet1/0`	Specifies the interface type and number and enters interface configuration mode.
Step 15	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding RED`	Configures the forwarding table.
Step 16	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 10.0.0.2 255.0.0.0`	Specifies the IP address and subnet mask for the interface.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] [**Example:** `Device(config)# route-map alpha permit ordering-seq`	Configures a route map and specifies how the packets are to be distributed. .
Step 19	**set ip vrf** *vrf-name* **next-hop verify-availability** *next-hop-address sequence* **track** *object* **Example:** `Device(config-route-map)# set ip vrf RED next-hop verify-availability 192.168.23.2 1 track 1`	Configures policy routing to verify the reachability of the next hop of a route map before the router performs policy routing to that next hop.
Step 20	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and returns to global configuration mode.
Step 21	**interface** *type number* **Example:** `Device(config)# interface Ethernet0/0`	Specifies the interface type and number and enters interface configuration mode.
Step 22	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding RED`	Configures the forwarding table.
Step 23	**ip policy route-map** *map-tag* **Example:** `Device(config-if)# ip policy route-map test02`	Identifies a route map to use for policy routing on an interface.

	Command or Action	Purpose
Step 24	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 192.168.10.2 255.255.255.0`	Specifies the IP address and subnet mask for the interface.
Step 25	**end** **Example:** `Device(config-if)# exit`	Returns to privileged EXEC mode.

Configuring PBR Next-Hop Verify Availability for Inherited IPv6 VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *vpn-route-distinguisher*
5. **route-target export** *route-target-ext-community*
6. **route-target import** *route-target-ext-community*
7. **exit**
8. **ip sla** *operation-number*
9. **icmp-echo** *destination-ip-address*
10. **vrf** *vrf-name*
11. **exit**
12. **ip sla schedule** *operation-number* **life forever start-time now**
13. **track** *object-number* **ip sla** *operation-number*
14. **interface** *type number*
15. **ip vrf forwarding** *vrf-name*
16. **ip address** *ip-address subnet-mask*
17. **ipv6 address** *ipv6-prefix*
18. **exit**
19. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] [
20. **set ipv6 vrf** *vrf-name* **next-hop verify-availability** *next-hop-address sequence* **track** *object*
21. **exit**
22. **interface** *type number*
23. **ip vrf forwarding** *vrf-name*
24. **ipv6 policy route-map** *map-tag*
25. **ip address** *ip-address subnet-mask*
26. **ipv6 address** *ipv6-prefix*
27. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf RED`	Configures an IP VPN routing and forwarding instance and enters VRF configuration mode.
Step 4	**rd** *vpn-route-distinguisher* **Example:** `Device(config-vrf)# rd 100:1`	Specifies the route distinguisher. The route distinguisher is either an autonomous system (AS) number or an IP address.
Step 5	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 100:1`	Creates a route-target extended community for a VRF and exports routing information from the target VPN extended community. The *route-target-ext-community* argument is either an AS number or an IP address.
Step 6	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 100:1`	Creates a route-target extended community for a VRF and imports routing information from the target VPN extended community. The *route-target-ext-community* argument is either an AS number or an IP address.
Step 7	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and returns to global configuration mode.
Step 8	**ip sla** *operation-number* **Example:** `Device(config)# ip sla 1`	Configures a Cisco IOS IP Service Level Agreements (SLAs) operation and enters IP SLA configuration mode.
Step 9	**icmp-echo** *destination-ip-address* **Example:** `Device(config-ip-sla)# icmp-echo 10.0.0.4`	Configures an IP SLAs Internet Control Message Protocol (ICMP) echo operation and enters ICMP echo configuration mode.
Step 10	**vrf** *vrf-name* **Example:** `Device(config-ip-sla-echo)# vrf RED`	Configures IP SLAs for a VRF instance.
Step 11	**exit** **Example:**	Exits ICMP echo configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-ip-sla-echo)# exit`	
Step 12	**ip sla schedule** *operation-number* **life forever start-time now** **Example:** `Device(config)# ip sla schedule 1 life forever start-time now`	Configures the scheduling parameters for a single Cisco IOS IP SLAs operation.
Step 13	**track** *object-number* **ip sla** *operation-number* **Example:** `Device(config)# track 1 ip sla 1`	Tracks the state of a Cisco IOS IP SLAs operation and enters tracking configuration mode.
Step 14	**interface** *type number* **Example:** `Device(config-track)# interface Ethernet1/0`	Specifies the interface type and number and enters interface configuration mode.
Step 15	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding RED`	Configures the forwarding table.
Step 16	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 10.0.0.2 255.0.0.0`	Specifies the IP address and subnet mask for the interface.
Step 17	**ipv6 address** *ipv6-prefix* **Example:** `Device(config-if)# ipv6 address 2001:DB8::/48`	Specifies the IPv6 prefix.
Step 18	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 19	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] [**Example:** `Device(config)# route-map alpha permit ordering-seq`	Configures a route map and specifies how the packets are to be distributed. .
Step 20	**set ipv6 vrf** *vrf-name* **next-hop verify-availability** *next-hop-address sequence* **track** *object* **Example:** `Device(config-route-map)# set ipv6 vrf RED next-hop verify-availability 2001:DB8:1::1 1 track 1`	Configures policy routing to verify the reachability of the next hop of a route map before the router performs policy routing to that next hop.

	Command or Action	Purpose
Step 21	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and returns to global configuration mode.
Step 22	**interface** *type number* **Example:** `Device(config)# interface Ethernet0/0`	Specifies the interface type and number and enters interface configuration mode.
Step 23	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding RED`	Configures the forwarding table.
Step 24	**ipv6 policy route-map** *map-tag* **Example:** `Device(config-if)# ipv6 policy route-map test02`	Identifies a route map to use for policy routing on an interface.
Step 25	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 192.168.10.2 255.255.255.0`	Specifies the IP address and subnet mask for the interface.
Step 26	**ipv6 address** *ipv6-prefix* **Example:** `Device(config-if)# ipv6 address 2001:DB8::/32`	Specifies the IPv6 prefix.
Step 27	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

Configuring PBR Next-Hop Verify Availability for Inter VRF

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *vpn-route-distinguisher*
5. **route-target export** *route-target-ext-community*
6. **ip vrf** *vrf-name*
7. **no rd** *vpn-route-distinguisher*
8. **rd** *vpn-route-distinguisher*
9. **route-target export** *route-target-ext-community*
10. **interface** *type number*

11. **ip vrf forwarding** *vrf-name*
12. **ip address** *ip-address subnet-mask*
13. **ip policy route-map** *map-tag*
14. **interface** *type number*
15. **ip vrf forwarding** *vrf-name*
16. **ip address** *ip-address subnet-mask*
17. **exit**
18. **ip route vrf** *vrf-name prefix mask interface-type interface-number ip-address*
19. **ip route vrf** *vrf-name prefix mask ip-address*
20. Repeat Step 19 to establish additional static routes.
21. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] [*sequence-name*
22. **match interface** *interface-type interface-number*
23. **set ip vrf** *vrf-name* **next-hop verify-availability** *next-hop-address sequence* **track** *object*
24. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf BLUE`	Configures an IP VPN routing and forwarding instance and enters VRF configuration mode.
Step 4	**rd** *vpn-route-distinguisher* **Example:** `Device(config-vrf)# rd 800:1`	Specifies the route distinguisher. The route distinguisher is either an autonomous system (AS) number or an IP address.
Step 5	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 800:1`	Creates a route-target extended community for a VRF and exports routing information from the target VPN extended community. The *route-target-ext-community* argument is either an AS number or an IP address.
Step 6	**ip vrf** *vrf-name* **Example:** `Device(config-vrf)# ip vrf BLUE`	Configures an IP VPN routing and forwarding instance.
Step 7	**no rd** *vpn-route-distinguisher* **Example:** `Device(config-vrf)# no rd 800:1`	Removes the specified route distinguisher.

	Command or Action	Purpose
Step 8	**rd** *vpn-route-distinguisher* **Example:** `Device(config-vrf)# rd 900:1`	Specifies the route distinguisher. The route distinguisher is either an AS number or an IP address.
Step 9	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 900:1`	Creates a route-target extended community for a VRF and exports routing information from the target VPN extended community. The *route-target-ext-community* argument is either an AS number or an IP address.
Step 10	**interface** *type number* **Example:** `Device(config-vrf)# interface Ethernet0/0`	Specifies the interface type and number and enters interface configuration mode.
Step 11	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding RED`	Configures the forwarding table.
Step 12	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 192.168.10.2 255.255.255.0`	Specifies the IP address and subnet mask for the interface.
Step 13	**ip policy route-map** *map-tag* **Example:** `Device(config-if)# ip policy route-map test00`	Identifies a route map to use for policy routing on an interface.
Step 14	**interface** *type number* **Example:** `Device(config-if)# interface Ethernet0/1`	Specifies the interface type and number.
Step 15	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding BLUE`	Configures the forwarding table.
Step 16	**ip address** *ip-address subnet-mask* **Example:** `Device(config-if)# ip address 192.168.21.1 255.255.255.0`	Specifies the IP address and subnet mask for the interface.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 18	**ip route vrf** *vrf-name prefix mask interface-type interface-number ip-address* **Example:**	Establishes static routes.

	Command or Action	Purpose
	`Device(config)# ip route vrf BLUE 192.168.10.1 255.255.255.255 Ethernet0/0 192.168.10.1`	
Step 19	**ip route vrf** *vrf-name prefix mask ip-address* **Example:** `Device(config)# ip route vrf BLUE 192.168.23.0 255.255.255.0 192.168.21.2`	Establishes static routes.
Step 20	Repeat Step 19 to establish additional static routes.	—
Step 21	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] [*sequence-name* **Example:** `Device(config)# route-map alpha permit ordering-seq`	Configures a route map and specifies how the packets are to be distributed..
Step 22	**match interface** *interface-type interface-number* **Example:** `Device(config-route-map)# match interface Ethernet0/0`	Distributes any routes that have their next hop as one of the specified interfaces.
Step 23	**set ip vrf** *vrf-name* **next-hop verify-availability** *next-hop-address sequence* **track** *object* **Example:** `Device(config-route-map)# set ip vrf BLUE next-hop verify-availability 192.168.23.2 1 track 1`	Configures policy routing to verify the reachability of the next hop of a route map of a VRF instance before the router performs policy routing to that next hop.
Step 24	**end** **Example:** `Device(config-route-map)# end`	Returns to privileged EXEC mode.

Configuration Examples for PBR Next-Hop Verify Availability for VRF

Example: Configuring PBR Next-Hop Verify Availability for Inherited IP VRF

```
Device> enable
Device# configure terminal
Device(config)# ip vrf RED
Device(config-vrf)# rd 100:1
Device(config-vrf)# route-target export 100:1
Device(config-vrf)# route-target import 100:1
Device(config-vrf)# exit
Device(config)# ip sla 1
Device(config-ip-sla)# icmp-echo 10.0.0.4
```

```
Device(config-ip-sla-echo)# vrf RED
Device(config-ip-sla-echo)# exit
Device(config)# ip sla schedule 1 life forever start-time now
Device(config)# track 1 ip sla 1
Device(config-track)# interface Ethernet0/0
Device(config-if)# ip vrf forwarding RED
Device(config-if)# ip address 10.0.0.2 255.0.0.0
Device(config-if)# exit
Device(config)# route-map test02 permit 10
Device(config-route-map)# set ip vrf RED next-hop verify-availability 192.168.23.2 1 track
 1
Device(config-route-map)# interface Ethernet0/0
Device(config-if)# ip vrf forwarding RED
Device(config-if)# ip policy route-map test02
Device(config-if)# ip address 192.168.10.2 255.255.255.0
Device(config-if)# end
```

Example: Configuring PBR Next-Hop Verify Availability for Inherited IPv6 VRF

```
Device> enable
Device# configure terminal
Device(config)# ip vrf RED
Device(config-vrf)# rd 100:1
Device(config-vrf)# route-target export 100:1
Device(config-vrf)# route-target import 100:1
Device(config-vrf)# exit
Device(config)# ip sla 1
Device(config-ip-sla)# icmp-echo 10.0.0.4
Device(config-ip-sla-echo)# vrf RED
Device(config-ip-sla-echo)# exit
Device(config)# ip sla schedule 1 life forever start-time now
Device(config)# track 1 ip sla 1
Device(config-track)# interface Ethernet0/0
Device(config-if)# ip vrf forwarding RED
Device(config-if)# ip policy route-map test02
Device(config-if)# ip address 192.168.10.2 255.255.255.0
Device(config-if)# ipv6 address 2001:DB8::/32
Device(config-if)# interface Ethernet1/0
Device(config-if)# ip vrf forwarding RED
Device(config-if)# ip address 10.0.0.2 255.0.0.0
Device(config-if)# ipv6 address 2001:DB8::/48
Device(config-if)# exit
Device(config)# route-map test02 permit 10
Device(config-route-map)# set ipv6 vrf RED next-hop verify-availability 2001:DB8:1::1 1
track 1
Device(config-route-map)# end
```

Example: Configuring PBR Next-Hop Verify Availability for Inter VRF

```
Device> enable
Device# configure terminal
Device(config)# ip vrf BLUE
Device(config-vrf)# rd 800:1
Device(config-vrf)# route-target export 800:1
Device(config-vrf)# ip vrf BLUE
Device(config-vrf)# no rd 800:1
Device(config-vrf)# rd 900:1
```

```
Device(config-vrf)# route-target export 900:1
Device(config-vrf)# interface Ethernet0/0
Device(config-if)# ip vrf forwarding RED
Device(config-if)# ip address 192.168.10.2 255.255.255.0
Device(config-if)# ip policy route-map test00
Device(config-if)# interface Ethernet0/1
Device(config-if)# ip vrf forwarding BLUE
Device(config-if)# ip address 192.168.21.1 255.255.255.0
Device(config-if)# exit
Device(config)# ip route vrf blue 192.168.10.1 255.255.255.255 Ethernet0/0 192.168.10.1
Device(config)# ip route vrf blue 192.168.23.0 255.255.255.0 192.168.21.2
Device(config)# route-map test00 permit 10
Device(config-route-map)# match interface Ethernet0/0
Device(config-route-map)# set ip vrf blue next-hop verify-availability 192.168.23.2 1 track
 1
Device(config-route-map)# end
```

Additional References for PBR Next-Hop Verify Availability for VRF

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for PBR Next-Hop Verify Availability for VRF

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.

Feature Name	Releases	Feature Information
PBR Next-Hop Verify Availability for VRF		The PBR Next-Hop Verify Availability for VRF feature enables verification of next-hop availability for IPv4/IPv6 packets in virtual routing and forwarding (VRF) instances.