Configuring MPLS over GRE

This chapter describes how to configure a Virtual Private Network (VPN) generic routing encapsulation (GRE) tunnel for moving Multiprotocol Label Switching (MPLS) packets over a non-MPLS network.

This chapter includes the following sections:

Finding Feature Information

Your software release might not support all the features documented in this module. For the latest caveats and feature information, see the Bug Search Tool at https://tools.cisco.com/bugsearch/ and the release notes for your 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 “New and Changed Information” chapter or the Feature History table below.

Information About Configuring MPLS over GRE

This section includes the following topics:

PE-to-PE GRE Tunneling

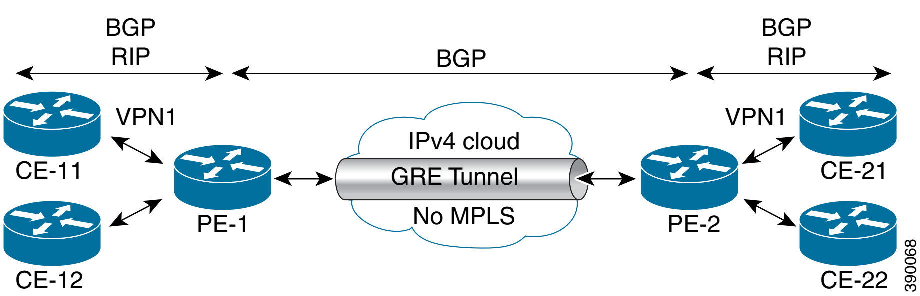
A provider-edge-to-provider-edge (PE-to-PE) tunnel provides a scalable way to connect multiple customer networks across a non-MPLS network. With this configuration, traffic that is destined to multiple customer networks is multiplexed through a single generic routing encapsulation (GRE) tunnel. A similar nonscalable alternative is to connect each customer network through separate GRE tunnels (for example, connecting one customer network to each GRE tunnel).

The PE devices assign virtual routing and forwarding (VRF) numbers to the customer edge (CE) devices on each side of the non-MPLS network. The PE devices use routing protocols such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), or Routing Information Protocol (RIP) to learn about the IP networks behind the CE devices. The routes to the IP networks behind the CE devices are stored in the VRF routing table of the associated CE device.

The PE device on one side of the non-MPLS network uses routing protocols (that operate within the non-MPLS network) to learn about the PE device on the other side of the non-MPLS network. The learned routes that are established between the PE devices are then stored in the main or default routing table. PE device on the other side of the network uses BGP to learn about the routes that are associated with the customer networks that are associated with the PE devices. These learned routes are not known to the non-MPLS network.

The following figure shows BGP defining a route to the BGP neighbor (the opposing PE device) through the GRE tunnel that spans the non-MPLS network. Because routes that are learned by the BGP neighbor include the GRE tunnel next hop, all customer network traffic is sent using the GRE tunnel.

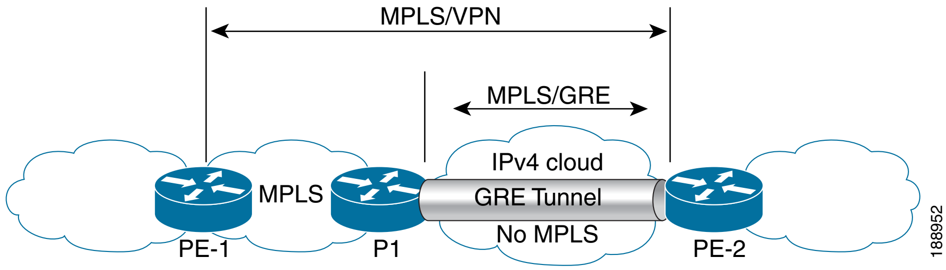
Figure 25-1 PE-to-PE GRE Tunnel



P-to-PE Tunneling

As shown in the figure below, the provider-to-provider-edge (P-to-PE) tunneling configuration provides a way to connect a PE device (P1) to a Multiprotocol Label Switching (MPLS) segment (PE-2) across a non-MPLS network. In this configuration, MPLS traffic that is destined to the other side of the non-MPLS network is sent through a single generic routing encapsulation (GRE) tunnel.

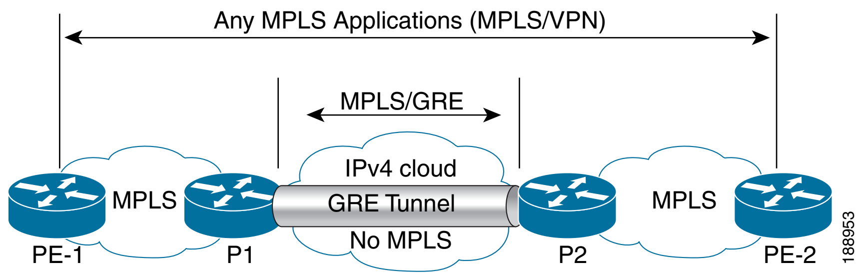
Figure 25-2 P-to-PE GRE Tunnel



P-to-P Tunneling

As shown in the figure below, the provider-to-provider (P-to-P) configuration provides a method of connecting two Multiprotocol Label Switching (MPLS) segments (P1 to P2) across a non-MPLS network. In this configuration, MPLS traffic that is destined to the other side of the non-MPLS network is sent through a single generic routing encapsulation (GRE) tunnel.

Figure 25-3 P-to-P Tunnel



Licensing Requirements for MPLS on GRE

The following table shows the licensing requirements for this feature:

Product	License Requirement
NX-OS	MPLS Layer 3 and Layer 2 VPNs require an MPLS license. For a complete explanation of the NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for Configuring MPLS over GRE

- Ensure that your MPLS VPN is configured and working properly.

Guidelines and Limitations for Configuring MPLS over GRE

- MPLS over GRE is supported on M1-Series and M2-Series I/O modules.

- MPLS over GRE is not supported on F3 series modules.

Layer 3 VPN MPLS over GRE does not support the following:

- Quality of service (QoS) service policies that are configured on the tunnel interface. QoS service policies are supported on the physical interface or subinterface.
- GRE options—Sequencing, checksum, and source route.
- IPv6 generic routing encapsulation (GRE).
- Advance features such as Carrier Supporting Carrier (CSC) and Interautonomous System (Inter-AS).
- GRE-based Layer 3 VPN does not interwork with MPLS or IP VPNs.
- GRE tunnel is supported only as a core link (PE-PE, PE-P, P-P, P-PE). A Provide-Edge to Customer-Edge (PE-CE) link is not supported.
- IPv6 VPN forwarding using GRE tunnels.
- Static route mapping to GRE tunnels.
- Bidirectional Forwarding Detection (BFD) with GRE tunnels.

Layer 2 VPLS over GRE has the following configuration guidelines and limitations:

- A VPLS instance must be configured on each Provider Edge (PE) device.
- Load balancing at the Virtual Private LAN Service (VPLS) ingress or at the core is not supported for flood or multicast traffic.
- Virtual circuit connection verification (VCCV) over flow aware transport of MPLS pseudowires (FAT PW) is not supported. The Interior Gateway Protocol (IGP) load balancing for VCCV is also unsupported.

Ethernet over MPLS over GRE has the following configuration guidelines and limitations:

- Multiple point-to-point tunnels can saturate the physical link with routing information if bandwidth is not configured correctly on a tunnel interface.
- A tunnel may have a recursive routing problem if routing is not configured accurately. The best path to a tunnel destination through the tunnel itself; therefore recursive routing causes the tunnel interface to flap. To avoid recursive routing problems, keep control-plane routing separate from tunnel routing by using the following methods:
  - Use a different autonomous system number or tag.
  - Use a different routing protocol.
  - Ensure that static routes are used to override the first hop (watch for routing loops).
- The following error is displayed when there is recursive routing to a tunnel destination:

```
%TUN-RECURDOWN Interface Tunnel 0 temporarily disabled due to recursive routing
```

Configuring MPLS over GRE

This section includes the following topics:

Configuring Layer 3 VPN Configuring MPLS over GRE

To configure a generic routing encapsulation (GRE) tunnel and create a virtual point-to-point link across the non-MPLS network, you must perform this task on the devices located at both ends of the GRE tunnel.

SUMMARY STEPS

1. **feature mpls**
2. **feature tunnel**
3. **configure terminal**
4. **interface tunnel** *tunnel-number*
5. **ip address** *ip-address ip-address-mask*
6. **mpls ip**
7. **tunnel source** *source-address*
8. **tunnel destination** *destination-address*
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**feature mpls** **Example** `switch# feature mpls`	Enables MPLS-related features.
Step 2	**feature tunnel** **Example:** `switch# feature tunnel`	Enables tunnel-related features.

Step 3	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 4	**interface tunnel** *tunnel-number*
	
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Enters interface configuration mode and creates a tunnel interface.
		
		The range for the *tunnel-number* argument is from 0 to 4095.

Step 5	**ip address** *ip-address mask*
	
	Example:
	`switch(config-if)# ip address 10.0.0.1`
	`255.255.255.0 3`
		Assigns an IP address to this tunnel interface.

Step 6	**ip address** *ip-address mask*
	
	Example:
	`switch(config-if)# ip address 10.0.0.1`
	`255.255.255.0 3`
		Assigns an IP address to this tunnel interface.

	Command	Purpose
Step 7	**`tunnel source`** *`source-address`* **Example:** `switch(config-if)# tunnel source 10.1.1.1`	Specifies the IP address of the tunnel source.
Step 8	**`tunnel destination`** *`destination-address`* **Example:** `switch(config-if)# tunnel source 10.1.1.2`	Specifies the IP address of the tunnel destination. • For provider edge (PE)-to-PE tunneling, configure tunnels with the same destination address.
Step 9	**`copy running-config startup-config`** **Example:** `switch(config-router-vrf-neighbor-af)# copy running-config startup-config`	(Optional) Copies the running configuration to the startup configuration.

Configuring Layer 2 VPN Configuring MPLS over GRE

Restrictions

You cannot have two tunnels using the same encapsulation mode with exactly the same source and destination addresses. The work around is to create a loopback interface and source packets from the loopback interface. This restriction is applicable only for generic routing encapsulation (GRE) tunnels.

SUMMARY STEPS

1. **configure terminal**
2. **interface loopback** *number*
3. **ip address** *ip-address mask*
4. **exit**
5. **interface tunnel** *number*
6. **tunnel mode gre**
7. **interface tunnel** *number*
8. **ip address** *ip-address mask*
9. **tunnel source** {*ip-address* | *type/number*}
10. **tunnel destination** {*hostname* | *ip-address*}
11. **mpls ip** {**propagate-ttl** | **ttl-expiration pop** [*labels*]}
12. **exit**
13. **ip route** *prefix mask interface-type interface-number*
14. **ip route** *prefix mask interface-type interface-number*
15. [**no**] **l2vpn vfi context** *context-name*
16. (Optional) **description** *description*
17. **vpn** *vpn-id*
18. **member peer** ip-address [*vc-id*] **encapsulation mpls**
19. **vlan configuration** *vlan-id*

20. **member vfi** *context-name*
21. (Optional) **copy running-config start-up config**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**`configure terminal`**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**`interface loopback`** *`number`*
	
	Example:
	`switch(config)# interface Loopback 0`
	`switch(config-if)#`
		Enters interface configuration mode and configures a loopback interface.
		• The range of the *number* argument is from 0 to 1023.

Step 3	**`ip address`** *`ip-address mask`* **Example:** `switch(config-if)# ip address 209.165.202.157 255.255.255.224`	Configures a primary address for this interface.

Step 4	**`exit`**
	
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits interface configuration mode and returns to global configuration mode.

Step 5	**`interface tunnel`** *`number`*
	
	Example:
	`switch(config)# interface tunnel 1`
	`switch(config-if)#`
		Enters interface configuration mode and configures a tunnel interface.
		• The range of the *number* argument is from 0 to 4095.
		• A tunnel should be independent of the endpoint physical interface type, such as TM, Gigabit, Packet over SONET (POS), and TenGigabit.
		• Up to 100 GRE tunnels are supported.

Step 6	**`tunnel mode gre`** **Example:** `switch(config-if)# tunnel mode gre`	Sets the encapsulation mode for the tunnel interface.
Step 7	**`interface tunnel`** *`number`* **Example:** `switch(config-if)# interface Tunnel 0`	Enters interface configuration mode and configures a tunnel interface. • The range of the *number* argument is from 0 to 4095.

	Command or Action	Purpose
Step 8	**ip address** *ip-address mask* **Example:** `switch(config-if)# ip address 209.165.200.225 255.255.255.224`	Configures a primary address for this interface.
Step 9	**tunnel source** {*ip-address* | *type***/***number*} **Example:** `switch(config-if)# tunnel source 192.0.0.2`	Sets the source address for a tunnel interface. • The source address is either an explicitly defined IP address or the IP address assigned to the specified interface. • GRE tunnel encapsulation and decapsulation for multicast packets are handled by the hardware. Each hardware-assisted tunnel must have a unique source. Hardware-assisted tunnels cannot share a source even if the destinations are different. You should use secondary addresses on loopback interfaces or create multiple loopback interfaces to ensure that the hardware-assisted tunnels do not share a source.
Step 10	**tunnel destination** {*hostname* | *ip-address*] **Example:** `switch(config-if)# tunnel destination 192.0.0.3`	Specifies the destination for the tunnel interface.
Step 11	**mpls ip** {**propagate-ttl** | **ttl-expiration-pop** [*labels*]} **Example:** `switch(config-if)# mpls ip propagate-ttl`	Controls the generation of the time-to-live (TTL) field in the Multiprotocol Label Switching (MPLS) header. • The **propagate-ttl** keyword enables MPLS forwarding along normally routed paths for the interface. • The **ttl-expiration-pop** keyword specifies that packets with an expired time-to-live (TTL) value are forwarded through the original label stack. • The *labels* argument is the maximum number of labels in the packet necessary for the packet to be forwarded by means of the global IP routing table.

Step 12	**exit**
	
	Example:
	`switch(config-if)# exit`
	`switch(config)#`
		Exits interface configuration mode and returns to global configuration mode.

Step 13	**ip route** *prefix mask interface-type interface-number* **Example:** `switch(config)# ip route 209.165.201.6 255.255.255.224 tunnel 1`	Creates a static route for routing packets for the designated network to the specified interface.

	Command or Action	Purpose
Step 14	**ip route** *prefix mask interface-type interface-number* **Example:** switch(config)# ip route 209.165.201.7 255.255.255.224 tunnel 2	Creates a static route for routing packets for the designated network to the specified interface.
Step 15	[**no**] **l2vpn vfi context** *context-name* **Example:** switch(config)# l2vpn vfi context example switch(config-l2vpn-vfi)#	Establishes a Layer 2 VPN (L2VPN) Virtual Forwarding Interface (VFI) context between two or more separate networks. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters. **Note** You can use the **no** form of this command to delete the context and the associated configuration.
Step 16	**description** *description* **Example:** switch(config-l2vpn-vfi)# description example	(Optional) Adds a description to the interface configuration. • The maximum range for the *description* argument is 254 alphanumeric characters.
Step 17	**vpn** *vpn-id* **Example:** switch(config-l2vpn-vfi)# vpn 100	Configures a VPN ID for a VFI context. • The emulated VCs bound to this Layer 2 VFI use this VPN ID for signaling. • The range of the *vpn-id* argument is from 1 to 4294967295.
Step 18	**member peer** *ip-address* [*vc-id*] **encapsulation mpls** **Example:** switch(config-l2vpn-vfi)# member peer 192.0.2.8 encapsulation mpls	Configures a dynamic pseudowire member under the VFI.
Step 19	**vlan configuration** *vlan-id* **Example:** switch(config-l2vpn-vfi)# vlan configuration 200 switch(config-vlan-config)#	Enters VLAN configuration mode and creates a VLAN ID.

	Command or Action	Purpose
Step 20	**member vfi** *context-name* **Example:** `switch(config-vlan-config)# member vfi example`	Adds the specified VFI context as a member of this VLAN. • The *context-name* argument is a unique per-interface identifier for this context. The maximum range is 100 alphanumeric, case-sensitive characters.
Step 21	**copy running-config startup-config** **Example:** `switch(config-vlan-config)# copy running-config startup-config`	(Optional) Saves this configuration change.

Verifying Configuring MPLS over GRE

To verify IP tunnel configuration information, perform one of the following tasks:

Command	Purpose
show interface tunnel *number*	Displays the configuration for the tunnel interface (MTU, protocol, transport, and VRF). Displays input and output packets, bytes, and packet rates.
show interface tunnel *number* **brief**	Displays the operational status, IP address, encapsulation type, and MTU of the tunnel interface.
show interface tunnel *number* **description**	Displays the configured description of the tunnel interface.
show interface tunnel *number* **status**	Displays the operational status of the tunnel interface.
show interface tunnel *number* **status err-disabled**	Displays the error disabled status of the tunnel interface.

Configuration Examples for Configuring MPLS over GRE

This section includes the following topics:

- Example: Configuring a GRE Tunnel That Spans a Non-MPLS Network, page 25-449
- Example: MPLS Configuration with PE-to-PE GRE Tunnel, page 25-450
- Example: MPLS Configuration with P-to-PE GRE Tunnel, page 25-453

Example: Configuring a GRE Tunnel That Spans a Non-MPLS Network

The following example shows how to configure a generic routing encapsulation (GRE) tunnel configuration that spans a non-MPLS network. This example shows the tunnel configuration on the provider edge (PE) devices (PE1 and PE2) located at both ends of the tunnel:

PE1 configuration

```
switch# configure terminal
switch(config)# interface Tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# mpls ip
switch(config-if)# ip address 10.1.1.1 255.255.255.0
switch(config-if)# tunnel source 10.0.0.1
switch(config-if)# tunnel destination 10.0.0.2
switch(config-if)# end
```

PE2 configuration

```
switch# configure terminal
switch(config)# interface Tunnel 1
switch(config-if)# tunnel mode gre ip
switch(config-if)# mpls ip
switch(config-if)# ip address 10.1.1.2 255.255.255.0
switch(config-if)# tunnel source 10.0.0.2
switch(config-if)# tunnel destination 10.0.0.1
switch(config-if)# end
```

Example: MPLS Configuration with PE-to-PE GRE Tunnel

The following example is for a basic PE-to-PE tunneling configuration that uses a generic routing encapsulation (GRE) tunnel to span a non-MPLS network:

PE1 configuration

```
feature-set mpls
feature mpls l3vpn
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature bgp

route-map allow permit 10

interface Tunnel0
/* description GRE tunnel */
  mpls ip
  ip address 10.1.1.1/24
  ip router ospf 100 area 0.0.0.0
  tunnel source Ethernet7/12
  tunnel destination 10.131.31.218
  no shutdown

interface Ethernet7/12
/* description Core facing interface */
  mpls ip
  ip address 10.131.31.205/30
  ip router rip 100
  no shutdown

interface loopback0
/* description Loopback for creating router sessions */
  ip address 10.131.31.1/32
  ipv6 address 1:1::1:1/128
  ip router ospf 100 area 0.0.0.0
```

```
interface loopback1
/*description Loopback for creating alternate router sessions */
  ip address 10.131.31.11/32
  ip router ospf 100 area 0.0.0.0

interface loopback11
/* description Loopback for testing vpn forwarding */
  vrf member vpn1
  ip address 1.1.1.1/24
  ipv6 address 11:11::11:1/120

vrf context vpn1
  rd 100:1
  address-family ipv4 unicast
    route-target import 100:1
    route-target export 100:1
  address-family ipv6 unicast
    route-target import 100:1
    route-target export 100:1

router bgp 100
  address-family ipv6 unicast
    redistribute direct route-map allow
    allocate-label all

  neighbor 10.131.31.2 remote-as 100
/* description VPNv4, VPNv6 */
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended

  neighbor 10.131.31.22 remote-as 100
/* description 6PE */
    update-source loopback1
    address-family ipv6 labeled-unicast
      send-community extended

  vrf vpn1
    address-family ipv4 unicast
      redistribute direct route-map allow
    address-family ipv6 unicast
      redistribute direct route-map allow

router ospf 100
router rip 100
```

PE2 configuration

```
feature-set mpls
feature mpls l3vpn
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature bgp

route-map allow permit 10

interface Tunnel0
/* description GRE tunnel */
  mpls ip
```

```
  ip address 10.1.1.2/24
  ip router ospf 100 area 0.0.0.0
  tunnel source Ethernet7/38
  tunnel destination 10.131.31.205
  no shutdown

interface Ethernet7/38
/* description Core facing interface */
  mpls ip
  ip address 10.131.31.218/30
  ip router rip 100
  no shutdown

interface loopback0
/*  description Loopback for creating router sessions */
  ip address 10.131.31.2/32
  ipv6 address 1:1::1:2/128
  ip router ospf 100 area 0.0.0.0

interface loopback1
/* description Loopback for creating alternate router sessions */
  ip address 10.131.31.22/32
  ip router ospf 100 area 0.0.0.0

interface loopback11
/* description Loopback for testing vpn forwarding */
  vrf member vpn1
  ip address 2.2.1.1/24
  ipv6 address 22:22::22:1/120

vrf context vpn1
  rd 100:1
  address-family ipv4 unicast
    route-target import 100:1
    route-target export 100:1
  address-family ipv6 unicast
    route-target import 100:1
    route-target export 100:1

router bgp 100
  address-family ipv6 unicast
    redistribute direct route-map allow
    allocate-label all

  neighbor 10.131.31.1 remote-as 100
/* description VPNv4, VPNv6 */
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended

  neighbor 10.131.31.11 remote-as 100
/* description 6PE */
    update-source loopback1
    address-family ipv6 labeled-unicast
      send-community extended

  vrf vpn1
    address-family ipv4 unicast
      redistribute direct route-map allow
    address-family ipv6 unicast
      redistribute direct route-map allow
```

```
router ospf 100
router rip 100
```

Example: MPLS Configuration with P-to-PE GRE Tunnel

The following example is for a basic P-to- PE tunneling configuration that uses a generic routing encapsulation (GRE) tunnel to span a non-MPLS network:

P configuration

```
feature-set mpls
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature mpls l3vpn

interface Tunnel0
/* description GRE tunnel */
mpls ip
ip address 10.1.1.1/24
ip router ospf 100 area 0.0.0.0
tunnel source Ethernet7/14
tunnel destination 10.131.31.205

interface Ethernet7/14
  mpls ip
  ip address 10.131.31.206/30
  ip router rip 100
  no shutdown

interface Ethernet7/36
  mpls ip
  ip address 10.131.31.217/30
  ip router rip 100
  no shutdown

interface loopback0
  ip address 10.131.31.10/32
  ip router ospf 100 area 0.0.0.0

router rip 100
router ospf 100
```

PE configuration

```
feature-set mpls
feature mpls l3vpn
feature mpls ldp
feature ospf
feature rip
feature tunnel
feature bgp

route-map allow permit 10

interface Tunnel0
/* description GRE tunnel */
  mpls ip
  ip address 10.1.1.2/24
```

```
  ip router ospf 100 area 0.0.0.0
  tunnel source Ethernet7/12
  tunnel destination 10.131.31.206
  no shutdown

interface Ethernet7/12
/* description Core facing interface */
  mpls ip
  ip address 10.131.31.205/30
  ip router rip 100
  no shutdown

interface loopback0
/* description Loopback for creating router sessions */
  ip address 10.131.31.1/32
  ipv6 address 1:1::1:1/128
  ip router ospf 100 area 0.0.0.0

interface loopback1
/* description Loopback for creating alternate router sessions */
  ip address 10.131.31.11/32
  ip router ospf 100 area 0.0.0.0

interface loopback11
/* description Loopback for testing vpn forwarding */
  vrf member vpn1
  ip address 1.1.1.1/24
  ipv6 address 11:11::11:1/120

vrf context vpn1
  rd 100:1
  address-family ipv4 unicast
    route-target import 100:1
    route-target export 100:1
  address-family ipv6 unicast
    route-target import 100:1
    route-target export 100:1

router bgp 100
  address-family ipv6 unicast
    redistribute direct route-map allow
    allocate-label all

  neighbor 10.131.31.2 remote-as 100
/* description VPNv4, VPNv6 */
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended

  neighbor 10.131.31.22 remote-as 100
/* description 6PE */
    update-source loopback1
    address-family ipv6 labeled-unicast
      send-community extended

  vrf vpn1
    address-family ipv4 unicast
      redistribute direct route-map allow
    address-family ipv6 unicast
      redistribute direct route-map allow

router ospf 100
```

```
router rip 100
```

Additional References for Configuring MPLS over GRE

This section includes the following topics:

- Related Documents, page 25-455
- MIBs <Optional: remove if not applicable>, page 25-455

Related Documents

Related Topic	Document Title
CLI commands	*Cisco Nexus 7000 Series NX-OS MPLS Command Reference*
Interface commands	*Cisco Nexus 7000 Series NX-OS Interface Command Reference*

MIBs <Optional: remove if not applicable>

MIBs	MIBs Link
• MPLS-L3VPN-STD-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/dc-os/mibs

Feature History for Layer 3 VPN Configuring MPLS over GRE

Table 25-1 lists the release history for this feature.

Table 25-1 Feature History for Layer 3 VPN Mpls over GRE

Feature Name	Releases	Feature Information
MPLS over GRE	6.2(2)	The MPLS over GRE feature provides a mechanism for tunneling Multiprotocol Label Switching (MPLS) packets over a non-MPLS network.