Implementing Generic Routing Encapsulation

Generic Routing Encapsulation (GRE) is a tunneling protocol developed by Cisco Systems that encapsulates a wide variety of network layer protocols inside virtual point-to-point links over an Internet Protocol internetwork.

Feature History for Configuring Link Bundling on Cisco IOS XR Software

Release	Modification
Release 4.3.0	These feature were supported on the Cisco ASR 9000 Series Aggregation Services Routers: • MPLS/L3VPNoGRE on ASR 9000 Enhanced Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700 • RSVP/TEoGRE on ASR 9000 Enhanced Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700 • VRF aware GRE on ASR 9000 Enhanced Ethernet Line Card and Cisco ASR 9000 Series SPA Interface Processor-700 • L2VPN (VPWS and VPLS) on GRE for ASR 9000 Enhanced Ethernet Line Card only

Contents

This chapter includes these sections:

Prerequisites for Configuring Generic Routing Encapsulation

Before configuring Link Bundling, be sure that these tasks and conditions are met:

- You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command.

  If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

Information About Generic Routing Encapsulation

To implement the GRE feature, you must understand these concepts:

- GRE Overview, page LSC-128
- GRE Features, page LSC-128

GRE Overview

Generic Routing Encapsulation (GRE) tunneling protocol provides a simple generic approach to transport packets of one protocol over another protocol by means of encapsulation.

GRE encapsulates a payload, that is, an inner packet that needs to be delivered to a destination network inside an outer IP packet. GRE tunnel endpoints send payloads through GRE tunnels by routing encapsulated packets through intervening IP networks. Other IP routers along the way do not parse the payload (the inner packet); they only parse the outer IP packet as they forward it towards the GRE tunnel endpoint. Upon reaching the tunnel endpoint, GRE encapsulation is removed and the payload is forwarded to it's ultimate destination.

MPLS networks provide VPN functionality by tunneling customer data through public networks using routing labels. Service Providers (SP) provide MPLS L3VPN, 6PE/6VPE and L2VPN services to their customers who have interconnected private networks.

MPLS and L3VPN are supported over regular interfaces on Cisco ASR 9000 Series Aggregation Services Routers. MPLS support is extended over GRE tunnels between routers as the provider core may not be fully MPLS aware.

GRE Features

Some of the supported features are:

- MPLS/L3VPN over GRE, page LSC-128
- 6PE/6VPE over GRE, page LSC-131

MPLS/L3VPN over GRE

The MPLS VPN over GRE feature provides a mechanism for tunneling Multiprotocol Label Switching (MPLS) packets over a non-MPLS network. This feature utilizes MPLS over generic routing encapsulation (MPLSoGRE) to encapsulate MPLS packets inside IP tunnels. The encapsulation of MPLS packets inside IP tunnels creates a virtual point-to-point link across non-MPLS networks.

L3VPN over GRE basically means encapsulating L3VPN traffic in GRE header and its outer IPv4 header with tunnel destination and source IP addresses after imposing zero or more MPLS labels, and transporting it across the tunnel over to the remote tunnel end point. The incoming packet can be a pure IPv4 packet or an MPLS packet. If the incoming packet is IPv4, the packet enters the tunnel through a VRF interface, and if the incoming packet is MPLS, then the packet enters through an MPLS interface. In the IPv4 case, before encapsulating in the outer IPv4 and GRE headers, a VPN label corresponding to the VRF prefix and any IGP label corresponding to the IGP prefix of the GRE tunnel destination is imposed on the packet. In the case of MPLS, the top IGP label is swapped with any label corresponding to the GRE tunnel destination address.

PE-to-PE Tunneling

The provider-edge-to-provider-edge (PE-to-PE) tunneling configuration 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 GRE tunnel.

Note A similar nonscalable alternative is to connect each customer network through separate GRE tunnels (for example, connecting one customer network to each GRE tunnel).

As shown in the Figure 8, the PE devices assign VPN routing and forwarding (VRF) numbers to the customer edge (CE) devices on each side of the non-MPLS network.

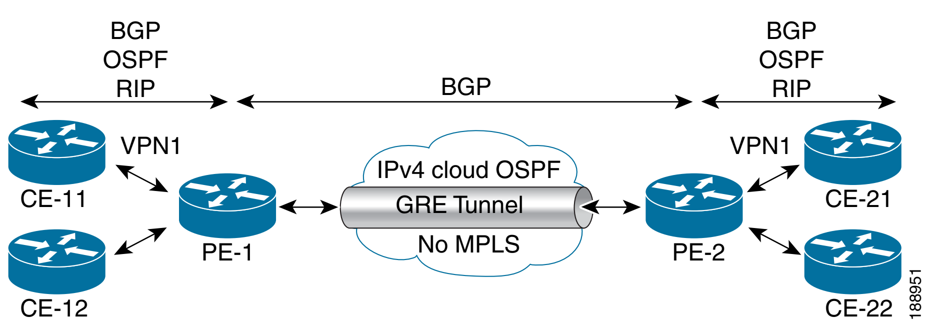
The PE devices use routing protocols such as Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), or Routing Information Protocol (RIP) to learn about the IP networks behind the CE devices. The routes to the IP networks behind the CE devices are stored in the associated CE device's VRF routing table.

The PE device on one side of the non-MPLS network uses the 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.

The opposing PE device uses BGP to learn about the routes that are associated with the customer networks that are behind the PE devices. These learned routes are not known to the non-MPLS network.

Figure 8 shows BGP defining a static 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.

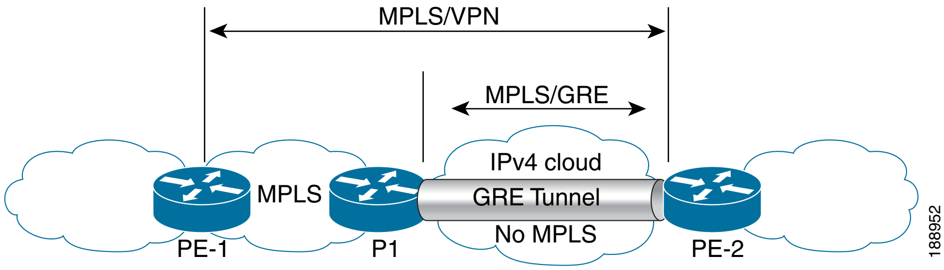
Figure 8 PE-to-PE Tunneling



P-to-PE Tunneling

As shown in Figure 9, the provider-to-provider-edge (P-to-PE) tunneling configuration provides a way to connect a PE device (P1) to an 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 GRE tunnel.

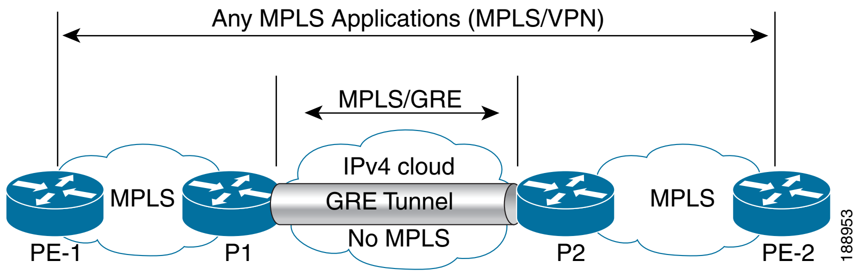
Figure 9 P-to-PE Tunneling



P-to-P Tunneling

As shown in Figure 10, the provider-to-provider (P-to-P) configuration provides a method of connecting two 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 GRE tunnel.

Figure 10 P-to-P Tunneling



6PE/6VPE

Service Providers (SPs) use a stable and established core with IPv4/MPLS backbone for providing IPv4 VPN services. The 6PE/6VPE feature facilitates SPs to offer IPv6 VPN services over this backbone without an IPv6 core. The provide edge (PE) routers run MP-iBGP (Multi-Protocol iBGP) to advertise v6 reachability and v6 label distribution. For 6PE, the labels are allocated per IPv6 prefix learnt from connected customer edge (CE) routers and for 6VPE, the PE router can be configured to allocate labels on a per-prefix or per-CE/VRF level.

6PE/6VPE over GRE

While IPv4/MPLS allows SPs to transport IPv6 traffic across IPv4 core (IPv6 unaware), MPLS over GRE allows MPLS traffic to be tunneled through MPLS unaware networks. These two features together facilitate IPv6 traffic to be transported across IPv6 as well as MPLS unaware core segments. Only the PE routers need to be aware of MPLS and IPv6 (Dual stack).

The 6PE/6VPE over GRE feature allows the use of IPv4 GRE tunnels to provide IPv6 VPN over MPLS functionality to reach the destination v6 prefixes via the BGP next hop through MPLS & IPv6 unaware core.

MPLS Forwarding

When IPv6 traffic is received from one customer site, the ingress PE device uses MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop. The ingress PE device prefixes the IPv6 packets with the outer and inner labels before placing the packet on the egress interface.

Under normal operation, a P device along the forwarding path does not lookup the frame beyond the first label. The P device either swaps the incoming label with an outgoing one or removes the incoming label if the next device is a PE device. Removing the incoming label is called penultimate hop popping. The remaining label (BGP label) is used to identify the egress PE interface toward the customer site. The label also hides the protocol version (IPv6) from the last P device, which it would otherwise need to forward an IPv6 packet.

A P device is ignorant of the IPv6 VPN routes. The IPv6 header remains hidden under one or more MPLS labels. When the P device receives an MPLS-encapsulated IPv6 packet that cannot be delivered, it has two options. If the P device is IPv6 aware, it exposes the IPv6 header, builds an Internet Control Message Protocol (ICMP) for IPv6 message, and sends the message, which is MPLS encapsulated, to the source of the original packet. If the P device is not IPv6 aware, it drops the packet.

6PE/6VPE over GRE

As discussed earlier, 6PE/6VPE over GRE basically means enabling IPv6/IPv6 VPN over MPLS over GRE.

The ingress PE device uses IPv4 generic routing encapsulation (GRE) tunnels combined with 6PE/6VPE over MPLS to tunnel IPv6 VPN packets over the backbone toward the egress PE device identified as the BGP next hop.

The PE devices establish MP-iBGP sessions and MPLS LDP sessions just as in the case of 6PE/6VPE. The difference here is that these sessions are established over GRE tunnels, which also means that the PEs are just one IGP hop away. The P routers in the tunnel path only need to forward the traffic to the tunnel destination, which is an IPv4 address.

This is how the IPv6 LSP is setup for label switching the IPv6 traffic:

- After the LDP and BGP sessions are established, the PEs exchange IPv6 prefixes that they learn from the CEs and the corresponding IPv6 labels, just as in the case of IPv4 VPN.

- The IPv6 labels occupy the inner most position in the label stack.
- The IPv4 labels corresponding to the PE IPv4 addresses occupy the outer position in the stack.
- When IPv6 traffic needs to be forwarded from PE1 to PE2, the outer PE2 IPv4 label is used to label switch the traffic to PE2, and the inner IPv6 label is used to send the packet out of the interface connected to the CE.

How to Configure Generic Routing Encapsulation

This section describes the tasks that are required to implement GRE:

- Configuring a GRE Tunnel, page LSC-132
- Configuring Global VRF, page LSC-134
- Configuring a VRF Interface, page LSC-136
- Configuring VRF Routing Protocol, page LSC-138
- Configuring IGP for Remote PE Reachability, page LSC-139
- Configuring LDP on GRE Tunnel, page LSC-141
- Configuring MP-iBGP to Exchange VPN-IPv4 Routes, page LSC-143

Configuring a GRE Tunnel

Perform this task to configure a GRE tunnel.

SUMMARY STEPS

1. **configure**
2. **interface tunnel-ip** *number*
3. **ipv4 address** *ipv4-address mask*
4. **tunnel source type** *path-id*
5. **tunnel destination** *ip-address*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface tunnel-ip** *number* **Example:** `RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 4000`	Enters tunnel interface configuration mode. • number is the number associated with the tunnel interface.
Step 3	**ipv4 address** *ipv4-address subnet-mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 10.1.1.1 255.255.255.0`	Specifies the IPv4 address and subnet mask for the interface. • ipv4-address specifies the IP address of the interface. • subnet-mask specifies the subnet mask of the interface.
Step 4	**tunnel source** *type path-id* **Example:** `RP/0/RSP0/CPU0:router(config-if)# tunnel source TenGigE0/2/0/1`	Specifies the source of the tunnel interface.

	Command or Action	Purpose

Step 5	**`tunnel destination`** *`ip-address`*
	
	`Example:`
	`RP/0/RSP0/CPU0:router(config-if)# tunnel destination 145.12.5.2`
		Defines the tunnel destination.

Step 6	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Global VRF

Perform this task to configure a global VRF.

SUMMARY STEPS

1. **configure**
2. **vrf** *vrf-name*
3. **address-family { ipv4 | ipv6 } unicast**
4. **import route-target** [*as-number:nn* | *ip-address:nn*]
5. **export route-target** [*as-number:nn* | *ip-address:nn*]
6. **exit**
7. **exit**
8. **router bgp** *as-number*
9. **vrf** *vrf-name*
10. **rd {as-number:nn | ip-address:nn | auto}**

11. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config)# vrf vpn1`	Configures a VRF instance.
Step 3	**address-family { ipv4 | ipv6 } unicast** **Example:** `RP/0/RSP0/CPU0:router(config-vrf)# address-family { ipv4 | ipv6 } unicast`	Specifies either the IPv4 or IPv6 address family and enters address family configuration submode.
Step 4	**import route-target** [*as-number:nn* | *ip-address:nn*] **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# import route-target 2:1`	Specifies a list of route target (RT) extended communities. Only prefixes that are associated with the specified import route target extended communities are imported into the VRF.
Step 5	**export route-target** [*as-number:nn* | *ip-address:nn*] **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# export route-target 1:1`	Specifies a list of route target extended communities. Export route target communities are associated with prefixes when they are advertised to remote PEs. The remote PEs import them into VRFs which have import RTs that match these exported route target communities.
Step 6	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-vrf-af)# exit`	Exits VRF address family configuration mode and returns the router to VRF configuration mode.
Step 7	**exit** **Example:** `RP/0/RSP0/CPU0:router(config-vrf)# exit`	Exits VRF configuration mode and returns the router to global configuration mode.
Step 8	**router bgp** *as-number* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 1`	Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

	Command or Action	Purpose
Step 9	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# vrf vpn1`	Configures a VRF instance.
Step 10	**rd {as-number:nn | ip-address:nn | auto}** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-vrf)#rd auto`	Configures the route distinguisher.

Step 11	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-bgp-vrf)# end`
	or
	`RP/0/RSP0/CPU0:router(config-bgp-vrf)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring a VRF Interface

Perform this task to configure a VRF interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type interface-path-id*
3. **vrf** *vrf-name*
4. **ipv4 address** *ipv4-address mask*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config)# interface tunnel-ip 100`	Enters interface configuration mode.
Step 3	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-if)# vrf vrf_A`	Configures a VRF instance and enters VRF configuration mode.
Step 4	**ipv4 address** *ipv4-address mask* **Example:** `RP/0/RSP0/CPU0:router(config-if)# ipv4 address 192.168.1.27 255.255.255.0`	Configures a primary IPv4 address for the specified interface.

Step 5	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring VRF Routing Protocol

Perform this task to configure the VRF routing protocol.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **vrf** *vrf-name*
4. **router-id** {*router-id* | *type interface-path-id*}
5. **area** *area-id*
6. **interface** *type interface-path-id*
7. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router ospf** *process-name* **Example:** `RP/0/RSP0/CPU0:router(config)# router ospf 109`	Enters OSPF configuration mode allowing you to configure the OSPF routing process.
Step 3	**vrf** *vrf-name* **Example:** `RP/0/RSP0/CPU0:router(config-ospf)# vrf vrf_1`	Configures a VPN routing and forwarding (VRF) instance and enters VRF configuration mode for OSPF routing.
Step 4	**router-id** {*router-id* | type interface-path-id} **Example:** `RP/0/RSP0/CPU0:router(config-ospf-vrf)# router-id 172.20.10.10`	Configures the router ID for the OSPF routing process.
Step 5	**area** *area-id* **Example:** `RP/0/RSP0/CPU0:router(config-ospf-vrf)# area 0`	Configures the OSPF area as area 0.

	Command or Action	Purpose
Step 6	**interface** type interface-path-id **Example:** `RP/0/RSP0/CPU0:router(config-ospf-vrf-ar)# interface GigabitEthernet 0/3/0/0`	Associates interface GigabitEthernet 0/3/0/0 with area 0.

Step 7	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-ospf-vrf-ar)# end`
	or
	`RP/0/RSP0/CPU0:router(config-ospf-vrf-ar)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring IGP for Remote PE Reachability

Perform this task to configure IGP for remote PE reachability.

SUMMARY STEPS

1. **configure**
2. **router ospf** *process-name*
3. **router-id** {*router-id*}
4. **area** *area-id*
5. **interface tunnel-ip** *number*
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**router ospf** *process-name* **Example:** `RP/0/RSP0/CPU0:router(config)# router ospf 1`	Enables OSPF routing for the specified routing process and places the router in router configuration mode.
Step 3	**router-id** {*router-id*} **Example:** `RP/0/RSP0/CPU0:router(config-ospf)# router-id 1.1.1.1`	Configures a router ID for the OSPF process. **Note** We recommend using a stable IP address as the router ID.
Step 4	**area** *area-id* Example: `RP/0/RSP0/CPU0:router(config-ospf)# area 0`	Enters area configuration mode and configures an area for the OSPF process.

	Command or Action	Purpose
Step 5	**`interface tunnel-ip`** *`number`* **Example:** `RP/0/RSP0/CPU0:router(config-ospf-ar)# interface tunnel-ip 4`	Enters tunnel interface configuration mode. • number is the number associated with the tunnel interface.

Step 6	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-ospf-ar-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-ospf-ar-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring LDP on GRE Tunnel

Perform this task to configure LDP on a GRE tunnel.

SUMMARY STEPS

1. **configure**
2. **mpls ldp**
3. **router-id** {*router-id*}
4. **interface tunnel-ip** *number*
5. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**`configure`** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**`mpls ldp`** **Example:** `RP/0/RSP0/CPU0:router(config)# mpls ldp`	Enables MPLS LDP configuration mode.
Step 3	**`router-id`** {*router-id*} **Example:** `RP/0/RSP0/CPU0:router(config-ldp)# router-id 1.1.1.1`	Configures a router ID for the OSPF process. **Note** We recommend using a stable IP address as the router ID.
Step 4	**`interface tunnel-ip`** *number* **Example:** `RP/0/RSP0/CPU0:router(config-ldp)# interface tunnel-ip 4`	Enters tunnel interface configuration mode. • number is the number associated with the tunnel interface.

Step 5	**`end`**
	or
	`commit`
	
	Example:
	`RP/0/RSP0/CPU0:router(config-ldp-if)# end`
	or
	`RP/0/RSP0/CPU0:router(config-ldp-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring MP-iBGP to Exchange VPN-IPv4 Routes

Perform this task to configure MP-iBGP to exchange VPN-IPv4 routes.

SUMMARY STEPS

1. **configure**
2. **router bgp** *process-name*
3. **router-id** *ip-address*
4. **neighbor** *ip-address*
5. **remote-as** *as-number*
6. **update-source** *type interface-path-id*
7. **address-family {vpnv4 | vpnv6} unicast**
8. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**`configure`** **Example:** `RP/0/RSP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**`router bgp`** *`as-number`* **Example:** `RP/0/RSP0/CPU0:router(config)# router bgp 1`	Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.
Step 3	**`router-id`** *`ip-address`* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# router-id 1.1.1.1`	Configures the local router with a specified router ID.
Step 4	**`neighbor`** *`ip-address`* **Example:** `RP/0/RSP0/CPU0:router(config-bgp)# neighbor 4.4.4.4`	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address as a BGP peer.
Step 5	**`remote-as`** *`as-number`* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)#remote-as 1`	Creates a neighbor and assigns a remote autonomous system number to it.

	Command or Action	Purpose
Step 6	**update-source** *type interface-path-id* **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)#update-source Loopback0`	Allows sessions to use the primary IP address from a specific interface as the local address when forming a session with a neighbor.
Step 7	**address-family {vpnv4 | vpnv6} unicast** **Example:** `RP/0/RSP0/CPU0:router(config-bgp-nbr)# address-family vpnv4 unicast`	Enters address family configuration submode for the specified address family.

Step 8	**end**
	or
	commit
	
	Example:
	`RP/0/RSP0/CPU0:router(config-bgp-nbr)# end`
	or
	`RP/0/RSP0/CPU0:router(config-bgp-nbr)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuration Examples for Generic Routing Encapsulation

This section provides examples to configure GRE:

- Configuring a GRE Tunnel: Example, page LSC-145
- Configuring Global VRF: Example, page LSC-145
- Configuring a VRF Interface: Example, page LSC-145
- Configuring VRF Routing Protocol: Example, page LSC-146
- Configuring IGP for Remote PE Reachability: Example, page LSC-146
- Configuring LDP on GRE Tunnel: Example, page LSC-146
- Configuring MP-iBGP to Exchange VPN-IPv4 Routes: Example, page LSC-146

Configuring a GRE Tunnel: Example

This example shows how to configure a GRE tunnel:

```
configure
  interface tunnel-ip1
   ipv4 address 12.0.0.1 255.255.255.0
   tunnel source Loopback0
   tunnel destination 200.200.200.1
end
```

Configuring Global VRF: Example

This example shows how to configure global VRF:

```
configure
 vrf VRF1
  address-family ipv4 unicast
  import route-target 120.1
  export route-target 120.2
exit
exit
  router bgp120
  vrf VRF1
  rd auto
end
```

Configuring a VRF Interface: Example

This example shows how to configure a VRF interface:

```
configure
 interface tunnel-ip 100
  vrf VRF1
  ipv4 address 1.1.1.1 255.255.255.0
end
```

Configuring VRF Routing Protocol: Example

This example shows how to configure VRF routing protocol:

```
configure
  router ospf109
   vrf VRF1
   router-id 172.20.10.10
   area0
   interface GigabitEthernet0/3/0/0
end
```

Configuring IGP for Remote PE Reachability: Example

This example shows how to configure IGP for remote provider edge (PE) reachability:

```
configure
  router ospf109
   router-id 172.20.10.10
   area0
   interface tunnel-ip1
end
```

Configuring LDP on GRE Tunnel: Example

This example shows how to configure LDP on a GRE tunnel:

```
configure
 mpls ldp
 router-id 172.20.10.10
  interface tunnel-ip1
 end
```

Configuring MP-iBGP to Exchange VPN-IPv4 Routes: Example

This example shows how to configure MP-iBGP to exchange VPN-IPv4 routes:

```
configure
 router bgp100
  router-id 172.20.10.10
  neighbor 2.2.2.2 remote-as 100
  update-source Loopback0
  address-family vpnv4 unicast
end
```

Additional References

For additional information related to implementing VPLS, refer to these:

Related Documents

Related Topic	Document Title
Cisco IOS XR L2VPN commands	*Point to Point Layer 2 Services Commands* module in the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*
MPLS VPLS-related commands	*Multipoint Layer 2 Services Commands* module in the *Cisco ASR 9000 Series Aggregation Services Router L2VPN and Ethernet Services Command Reference*
Getting started material	*Cisco ASR 9000 Series Aggregation Services Router Getting Started Guide*
Traffic storm control on VPLS bridges	*Traffic Storm Control under VPLS Bridges on Cisco ASR 9000 Series Routers* module in the *Cisco ASR 9000 Series Aggregation Services Router System Security Configuration Guide*
Layer 2 multicast on VPLS bridges	*Layer 2 Multicast Using IGMP Snooping* module in the *Cisco ASR 9000 Series Aggregation Services Router Multicast Configuration Guide*

Standards

Standards[1]	Title
draft-ietf-l2vpn-vpls-ldp-09	*Virtual Private LAN Services Using LDP*

1. Not all supported standards are listed.

MIBs

MIBs	MIBs Link
—	To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at this URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 2784	*Generic Routing Encapsulation (GRE)*
RFC 4448	*Encapsulation Methods for Transport of Ethernet over MPLS Networks*, April 2006
RFC 4762	*Virtual Private LAN Service (VPLS) Using Label Distribution Protocol (LDP) Signaling*

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport