IP Routing: BGP Configuration Guide, Cisco IOS XE 17 (Cisco ASR 900 Series)

First Published: 2017-11-01

Last Modified: 2023-07-14

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

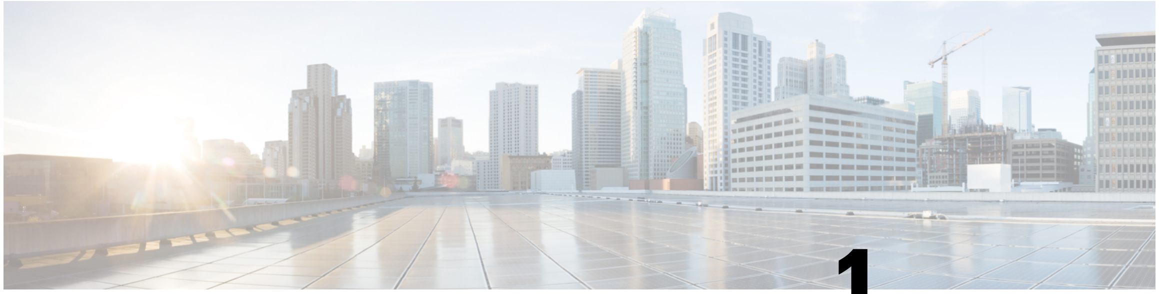
All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

CONTENTS

CHAPTER 1

CHAPTER 2



CHAPTER 1

Feature History

The following table lists the new and modified features supported in the IP Routing: BGP Configuration Guide in Cisco IOS XE 17 releases.

Feature	Description
Cisco IOS XE Bengaluru 17.4.1	
Convergence Enhancements for L3VPN Traffic over BGP PIC Edge	The BGP PIC feature is enhanced to achieve the sub-second convergence for greater than 2000 VPNv4/v6 prefixes using the hardware assisted PIC Edge functionality. Prior to this release, the BGP PIC Edge convergence could only be observed in seconds. To enable the feature, use the following new command: **platform l3vpn pic-enable**

CHAPTER 2

BGP 4

BGP is an interdomain routing protocol designed to provide loop-free routing between separate routing domains that contain independent routing policies (autonomous systems).

Information About BGP 4

BGP Version 4 Functional Overview

BGP is an interdomain routing protocol designed to provide loop-free routing links between organizations. BGP is designed to run over a reliable transport protocol; it uses TCP (port 179) as the transport protocol because TCP is a connection-oriented protocol. The destination TCP port is assigned 179, and the local port is assigned a random port number. Cisco software supports BGP version 4 and it is this version that has been used by Internet service providers (ISPs) to help build the Internet. RFC 1771 introduced and discussed a number of new BGP features to allow the protocol to scale for Internet use. RFC 2858 introduced multiprotocol extensions to allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families, including IPv4, IPv6, and CLNS.

BGP is mainly used to connect a local network to an external network to gain access to the Internet or to connect to other organizations. When connecting to an external organization, external BGP (eBGP) peering sessions are created. Although BGP is referred to as an exterior gateway protocol (EGP), many networks within an organization are becoming so complex that BGP can be used to simplify the internal network used within the organization. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

BGP uses a path-vector routing algorithm to exchange network reachability information with other BGP-speaking networking devices. Network reachability information is exchanged between BGP peers in routing updates. Network reachability information contains the network number, path-specific attributes, and the list of autonomous system numbers that a route must transit to reach a destination network. This list is contained in the AS-path attribute. BGP prevents routing loops by rejecting any routing update that contains the local autonomous system number because this indicates that the route has already traveled through that autonomous system and a loop would therefore be created. The BGP path-vector routing algorithm is a combination of the distance-vector routing algorithm and the AS-path loop detection.

BGP selects a single path, by default, as the best path to a destination host or network. The best path selection algorithm analyzes path attributes to determine which route is installed as the best path in the BGP routing table. Each path carries well-known mandatory, well-known discretionary, and optional transitive attributes that are used in BGP best path analysis. Cisco software provides the ability to influence BGP path selection by altering some of these attributes using the command-line interface (CLI.) BGP path selection can also be influenced through standard BGP policy configuration. For more details about using BGP to influence path selection and configuring BGP policies to filter traffic, see the "BGP 4 Prefix Filter and Inbound Route Maps" module and the "BGP Prefix-Based Outbound Route Filtering" module.

BGP uses the best-path selection algorithm to find a set of equally good routes. These routes are the potential multipaths. In Cisco IOS Release 12.2(33)SRD and later releases, when there are more equally good multipaths available than the maximum permitted number, the oldest paths are selected as multipaths.

BGP can be used to help manage complex internal networks by interfacing with Interior Gateway Protocols (IGPs). Internal BGP can help with issues such as scaling the existing IGPs to match the traffic demands while maintaining network efficiency.

Note BGP requires more configuration than other routing protocols and the effects of any configuration changes must be fully understood. Incorrect configuration can create routing loops and negatively impact normal network operation.

BGP Router ID

BGP uses a router ID to identify BGP-speaking peers. The BGP router ID is a 32-bit value that is often represented by an IPv4 address. By default, the Cisco software sets the router ID to the IPv4 address of a loopback interface on the router. If no loopback interface is configured on the device, the software chooses the highest IPv4 address configured on a physical interface of the device to represent the BGP router ID. The BGP router ID must be unique to the BGP peers in a network.

BGP-Speaker and Peer Relationships

A BGP-speaking device does not discover another BGP-speaking device automatically. A network administrator usually manually configures the relationships between BGP-speaking devices. A peer device is a BGP-speaking device that has an active TCP connection to another BGP-speaking device. This relationship between BGP devices is often referred to as a neighbor, but because this can imply the idea that the BGP devices are directly connected with no other device in between, the term *neighbor* will be avoided whenever possible in this document. A BGP speaker is the local device, and a peer is any other BGP-speaking network device.

When a TCP connection is established between peers, each BGP peer initially exchanges all its routes—the complete BGP routing table—with the other peer. After this initial exchange, only incremental updates are sent when there has been a topology change in the network, or when a routing policy has been implemented or modified. In the periods of inactivity between these updates, peers exchange special messages called keepalives.

A BGP autonomous system is a network that is controlled by a single technical administration entity. Peer devices are called external peers when they are in different autonomous systems and internal peers when they are in the same autonomous system. Usually, external peers are adjacent and share a subnet; internal peers may be anywhere in the same autonomous system.

BGP Peer Session Establishment

When a BGP routing process establishes a peering session with a peer, it goes through the following state changes:

- Idle—The initial state that the BGP routing process enters when the routing process is enabled or when the device is reset. In this state, the device waits for a start event, such as a peering configuration with a remote peer. After the device receives a TCP connection request from a remote peer, the device initiates another start event to wait for a timer before starting a TCP connection to a remote peer. If the device is reset, the peer is reset and the BGP routing process returns to the Idle state.
- Connect—The BGP routing process detects that a peer is trying to establish a TCP session with the local BGP speaker.
- Active—In this state, the BGP routing process tries to establish a TCP session with a peer device using the ConnectRetry timer. Start events are ignored while the BGP routing process is in the Active state. If the BGP routing process is reconfigured or if an error occurs, the BGP routing process will release system resources and return to an Idle state.
- OpenSent—The TCP connection is established, and the BGP routing process sends an OPEN message to the remote peer, and transitions to the OpenSent state. The BGP routing process can receive other OPEN messages in this state. If the connection fails, the BGP routing process transitions to the Active state.
- OpenReceive—The BGP routing process receives the OPEN message from the remote peer and waits for an initial keepalive message from the remote peer. When a keepalive message is received, the BGP routing process transitions to the Established state. If a notification message is received, the BGP routing process transitions to the Idle state. If an error or configuration change occurs that affects the peering session, the BGP routing process sends a notification message with the Finite State Machine (FSM) error code and then transitions to the Idle state.
- Established—The initial keepalive is received from the remote peer. Peering is now established with the remote neighbor and the BGP routing process starts exchanging update message with the remote peer. The hold timer restarts when an update or keepalive message is received. If the BGP process receives an error notification, it will transition to the Idle state.

BGP Session Reset

Whenever the routing policy changes due to a configuration change, BGP peering sessions must be reset by using the **clear ip bgp** command. Cisco software supports the following three mechanisms to reset BGP peering sessions:

- Hard reset—A hard reset tears down the specified peering sessions including the TCP connection and deletes routes coming from the specified peer.
- Soft reset—A soft reset uses stored prefix information to reconfigure and activate BGP routing tables without tearing down existing peering sessions. Soft reconfiguration uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply new BGP policy without disrupting the network. Soft reconfiguration can be configured for inbound or outbound sessions.

- Dynamic inbound soft reset—The route refresh capability, as defined in RFC 2918, allows the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers. The route refresh capability does not store update information locally for nondisruptive policy changes. It instead relies on dynamic exchange with supporting peers. Route refresh must first be advertised through BGP capability negotiation between peers. All BGP devices must support the route refresh capability. To determine if a BGP device supports this capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the device supports the route refresh capability:

```
Received route refresh capability from peer.
```

The **bgp soft-reconfig-backup** command was introduced to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.

BGP Route Aggregation

BGP peers store and exchange routing information and the amount of routing information increases as more BGP speakers are configured. The use of route aggregation reduces the amount of information involved. Aggregation is the process of combining the attributes of several different routes so that only a single route is advertised. Aggregate prefixes use the classless interdomain routing (CIDR) principle to combine contiguous networks into one classless set of IP addresses that can be summarized in routing tables. Fewer routes now need to be advertised.

Two methods are available in BGP to implement route aggregation. You can redistribute an aggregated route into BGP or you can use a form of conditional aggregation. Basic route redistribution involves creating an aggregate route and then redistributing the routes into BGP. Conditional aggregation involves creating an aggregate route and then advertising or suppressing the advertising of certain routes on the basis of route maps, autonomous system set path (AS-SET) information, or summary information.

The **bgp suppress-inactive** command configures BGP to not advertise inactive routes to any BGP peer. A BGP routing process can advertise routes that are not installed in the routing information database (RIB) to BGP peers by default. A route that is not installed into the RIB is an inactive route. Inactive route advertisement can occur, for example, when routes are advertised through common route aggregation. Inactive route advertisements can be suppressed to provide more consistent data forwarding.

BGP Route Aggregation Generating AS_SET Information

AS_SET information can be generated when BGP routes are aggregated using the **aggregate-address** command. The path advertised for such a route is an AS_SET consisting of all the elements, including the communities, contained in all the paths that are being summarized. If the AS_PATHs to be aggregated are identical, only the AS_PATH is advertised. The ATOMIC-AGGREGATE attribute, set by default for the **aggregate-address** command, is not added to the AS_SET.

Routing Policy Change Management

Routing policies for a peer include all the configurations for elements such as a route map, distribute list, prefix list, and filter list that may impact inbound or outbound routing table updates. Whenever there is a change in the routing policy, the BGP session must be soft-cleared, or soft-reset, for the new policy to take effect. Performing inbound reset enables the new inbound policy configured on the device to take effect. Performing outbound reset causes the new local outbound policy configured on the device to take effect without resetting the BGP session. As a new set of updates is sent during outbound policy reset, a new inbound policy of the neighbor can also take effect. This means that after changing inbound policy, you must do an inbound reset on the local device or an outbound reset on the peer device. Outbound policy changes require an outbound reset on the local device or an inbound reset on the peer device.

There are two types of reset: hard reset and soft reset. The table below lists their advantages and disadvantages.

Table 1: Advantages and Disadvantages of Hard and Soft Resets

Type of Reset	Advantages	Disadvantages
Hard reset	No memory overhead.	The prefixes in the BGP, IP, and Forwarding Information Base (FIB) tables provided by the neighbor are lost. A hard reset is not recommended.
Outbound soft reset	No configuration, and no storing of routing table updates.	Does not reset inbound routing table updates.
Dynamic inbound soft reset	Does not clear the BGP session and cache. Does not require storing of routing table updates, and has no memory overhead.	Both BGP devices must support the route refresh capability. **Note** Does not reset outbound routing table updates.
Configured inbound soft reset (uses the **neighbor soft-reconfiguration** router configuration command)	Can be used when both BGP devices do not support the automatic route refresh capability. The **bgp soft-reconfig-backup** command was introduced to configure inbound soft reconfiguration for peers that do not support the route refresh capability.	Requires preconfiguration. Stores all received (inbound) routing policy updates without modification; is memory-intensive. Recommended only when absolutely necessary, such as when both BGP devices do not support the automatic route refresh capability. **Note** Does not reset outbound routing table updates.

Once you have defined two devices to be BGP neighbors, they will form a BGP connection and exchange routing information. If you subsequently change a BGP filter, weight, distance, version, or timer, or if you make a similar configuration change, you must reset BGP connections in order for the configuration change to take effect.

A soft reset updates the routing table for inbound and outbound routing updates. Cisco software supports soft reset without any prior configuration. This soft reset allows the dynamic exchange of route refresh requests and routing information between BGP devices, and allows the subsequent readvertisement of the respective outbound routing table. There are two types of soft reset:

- When soft reset is used to generate inbound updates from a neighbor, it is called dynamic inbound soft reset.
- When soft reset is used to send a new set of updates to a neighbor, it is called outbound soft reset.

To use soft reset without preconfiguration, both BGP peers must support the soft route refresh capability, which is advertised in the OPEN message sent when the peers establish a TCP session.

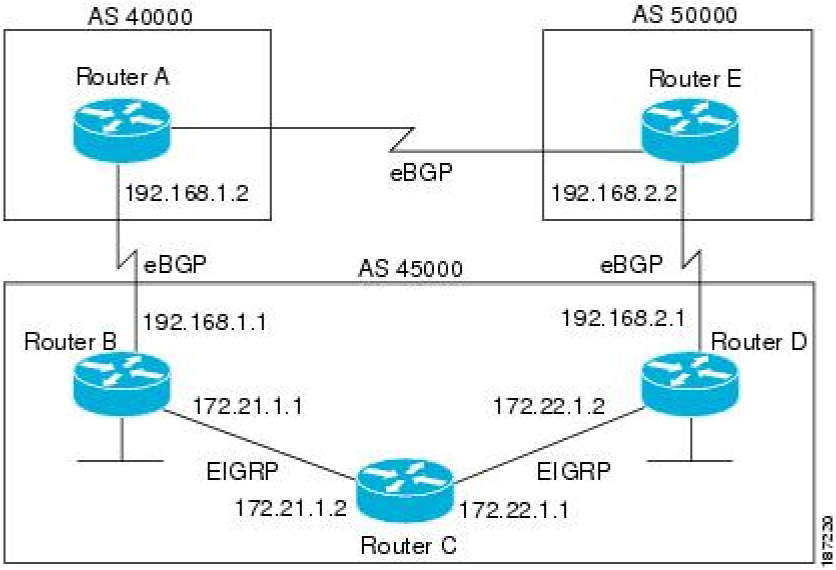
BGP Peer Groups

Often, in a BGP network, many neighbors are configured with the same update policies (that is, the same outbound route maps, distribute lists, filter lists, update source, and so on). Neighbors with the same update policies can be grouped into BGP peer groups to simplify configuration and, more importantly, to make configuration updates more efficient. When you have many peers, this approach is highly recommended.

BGP Backdoor Routes

In a BGP network topology with two border devices using eBGP to communicate to a number of different autonomous systems, using eBGP to communicate between the two border devices may not be the most efficient routing method. In the figure below, Router B as a BGP speaker will receive a route to Router D through eBGP, but this route will traverse at least two autonomous systems. Router B and Router D are also connected through an Enhanced Interior Gateway Routing Protocol (EIGRP) network (any IGP can be used here), and this route has a shorter path. EIGRP routes, however, have a default administrative distance of 90, and eBGP routes have a default administrative distance of 20, so BGP will prefer the eBGP route. Changing the default administrative distances is not recommended because changing the administrative distance may lead to routing loops. To cause BGP to prefer the EIGRP route, you can use the **network backdoor** command. BGP treats the network specified by the **network backdoor** command as a locally assigned network, except that it does not advertise the specified network in BGP updates. In the figure below, this means that Router B will communicate to Router D using the shorter EIGRP route instead of the longer eBGP route.

Figure 1: BGP Backdoor Route Topology



Restriction for BGP on the RSP3 Module

- Path MTU discovery is not supported on the RSP3 Module.

How to Configure BGP 4

Configuring a basic BGP network consists of a few required tasks and many optional tasks. A BGP routing process must be configured and BGP peers must be configured, preferably using the address family configuration model. If the BGP peers are part of a VPN network, the BGP peers must be configured using the IPv4 VRF address family task.

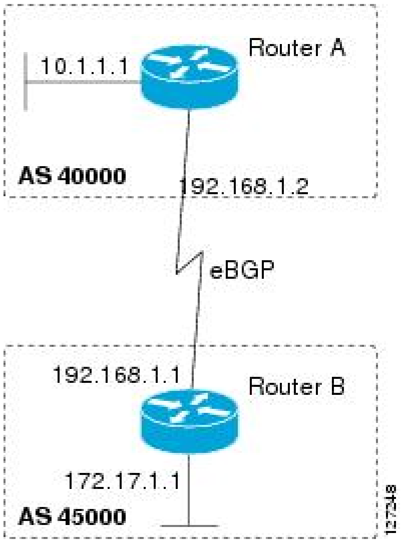
Configuring a BGP Routing Process

Perform this task to configure a BGP routing process. You must perform the required steps at least once to enable BGP. The optional steps here allow you to configure additional features in your BGP network. Several of the features, such as logging neighbor resets and immediate reset of a peer when its link goes down, are enabled by default but are presented here to enhance your understanding of how your BGP network operates.

Note A device that runs Cisco software can be configured to run only one BGP routing process and to be a member of only one BGP autonomous system. However, a BGP routing process and autonomous system can support multiple concurrent BGP address family and subaddress family configurations.

The configuration in this task is done at Router A in the figure below and would need to be repeated with appropriate changes to the IP addresses (for example, at Router B) to fully achieve a BGP process between the two devices. No address family is configured here for the BGP routing process, so routing information for the IPv4 unicast address family is advertised by default.

Figure 2: BGP Topology with Two Autonomous Systems



Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Configures a BGP routing process, and enters router configuration mode for the specified routing process.

- Use the *autonomous-system-number* argument to specify an integer, from 0 and 65534, that identifies the device to other BGP speakers.

Step 4 **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]

Example:

```
Device(config-router)# network 10.1.1.0 mask 255.255.255.0
```

(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table.

- For exterior protocols, the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.

Step 5 **bgp router-id** *ip-address*

Example:

```
Device(config-router)# bgp router-id 10.1.1.99
```

(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP.

- Use the *ip-address* argument to specify a unique router ID within the network.

Note Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.

Step 6 **timers bgp** *keepalive holdtime*

Example:

```
Device(config-router)# timers bgp 70 120
```

(Optional) Sets BGP network timers.

- Use the *keepalive* argument to specify the frequency, in seconds, with which the software sends keepalive messages to its BGP peer. By default, the keepalive timer is set to 60 seconds.
- Use the *holdtime* argument to specify the interval, in seconds, after which the software, having not received a keepalive message, declares a BGP peer dead. By default, the holdtime timer is set to 180 seconds.

Step 7 **bgp fast-external-fallover**

Example:

```
Device(config-router)# bgp fast-external-fallover
```

(Optional) Enables the automatic resetting of BGP sessions.

- By default, the BGP sessions of any directly adjacent external peers are reset if the link used to reach them goes down.

Step 8 **bgp log-neighbor-changes**

Example:

```
Device(config-router)# bgp log-neighbor-changes
```

(Optional) Enables logging of BGP neighbor status changes (up or down) and neighbor resets.

- Use this command for troubleshooting network connectivity problems and measuring network stability. Unexpected neighbor resets might indicate high error rates or high packet loss in the network and should be investigated.

Step 9 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 10 **show ip bgp** [*network*] [*network-mask*]

Example:

```
Device# show ip bgp
```

(Optional) Displays the entries in the BGP routing table.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following sample output from the **show ip bgp** command shows the BGP routing table for Router A in the figure above after this task has been configured on Router A. You can see an entry for the network 10.1.1.0 that is local to this autonomous system.

```
BGP table version is 12, local router ID is 10.1.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      0.0.0.0                  0         32768 i
```

Troubleshooting Tips

Use the **ping** command to check basic network connectivity between the BGP routers.

Configuring a BGP Peer

Perform this task to configure BGP between two IPv4 devices (peers). The address family configured here is the default IPv4 unicast address family, and the configuration is done at Router A in the figure above. Remember to perform this task for any neighboring devices that are to be BGP peers.

Before you begin

Before you perform this task, perform the “Configuring a BGP Routing Process” task.

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, such as IPv6 prefixes.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.1 remote-as 45000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 unicast
```

Specifies the IPv4 address family and enters address family configuration mode.

- The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command.
- The **multicast** keyword specifies IPv4 multicast address prefixes.
- The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.

Step 6 **neighbor** *ip-address* **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.1.1 activate
```

Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local device.

Step 7 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and returns to privileged EXEC mode.

Step 8 **show ip bgp** [*network*] [*network-mask*]

Example:

```
Device# show ip bgp
```

(Optional) Displays the entries in the BGP routing table.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 9 **show ip bgp neighbors** [*neighbor-address*]

Example:

```
Device(config-router-af)# show ip bgp neighbors 192.168.2.2
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following sample output from the **show ip bgp** command shows the BGP routing table for Router A in the figure above after this task has been configured on Router A and Router B. You can now see an entry for the network 172.17.1.0 in autonomous system 45000.

```
BGP table version is 13, local router ID is 10.1.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      0.0.0.0                  0         32768 i
*> 172.17.1.0/24    192.168.1.1              0             0 45000 i
```

The following sample output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.1.1 of Router A in the figure above after this task has been configured on Router A:

```
BGP neighbor is 192.168.1.1,  remote AS 45000, external link
  BGP version 4, remote router ID 172.17.1.99
  BGP state = Established, up for 00:06:55
  Last read 00:00:15, last write 00:00:15, hold time is 120, keepalive intervals
  Configured hold time is 120,keepalive interval is 70 seconds, Minimum holdtims
  Neighbor capabilities:
    Route refresh: advertised and received (old & new)
    Address family IPv4 Unicast: advertised and received
  Message statistics:
    InQ depth is 0
    OutQ depth is 0
                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                1          2
    Keepalives:            13         13
    Route Refresh:          0          0
    Total:                 15         16
  Default minimum time between advertisement runs is 30 seconds
 For address family: IPv4 Unicast
  BGP table version 13, neighbor version 13/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
                                 Sent       Rcvd
  Prefix activity:               ----       ----
    Prefixes Current:               1          1 (Consumes 52 bytes)
    Prefixes Total:                 1          1
    Implicit Withdraw:              0          0
    Explicit Withdraw:              0          0
    Used as bestpath:             n/a          1
    Used as multipath:            n/a          0
                                   Outbound    Inbound
  Local Policy Denied Prefixes:    --------    -------
    AS_PATH loop:                       n/a          1
    Bestpath from this peer:              1        n/a
    Total:                                1          1
  Number of NLRIs in the update sent: max 0, min 0
  Connections established 1; dropped 0
  Last reset never
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 192.168.1.2, Local port: 179
Foreign host: 192.168.1.1, Foreign port: 37725
Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)
Event Timers (current time is 0x12F4F2C):
Timer          Starts    Wakeups            Next
Retrans            14          0             0x0
TimeWait            0          0             0x0
AckHold            13          8             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0
iss:  165379618  snduna:  165379963  sndnxt:  165379963     sndwnd:  16040
irs: 3127821601  rcvnxt: 3127821993  rcvwnd:      15993  delrcvwnd:    391
SRTT: 254 ms, RTTO: 619 ms, RTV: 365 ms, KRTT: 0 ms
minRTT: 12 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: passive open, nagle, gen tcbs
IP Precedence value : 6
Datagrams (max data segment is 1460 bytes):
Rcvd: 20 (out of order: 0), with data: 15, total data bytes: 391
Sent: 22 (retransmit: 0, fastretransmit: 0, partialack: 0, Second Congestion: 04
```

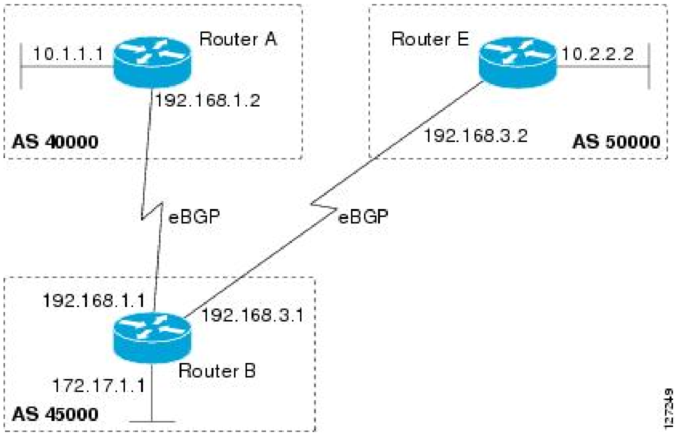
Troubleshooting Tips

Use the **ping** command to verify basic network connectivity between the BGP devices.

Configuring a BGP Peer for the IPv4 VRF Address Family

Perform this optional task to configure BGP between two IPv4 devices (peers) that must exchange IPv4 VRF information because they exist in a VPN. The address family configured here is the IPv4 VRF address family, and the configuration is done at Router B in the figure below with the neighbor 192.168.3.2 at Router E in autonomous system 50000. Remember to perform this task for any neighboring devices that are to be BGP IPv4 VRF address family peers.

Figure 3: BGP Topology for IPv4 VRF Address Family



Before you begin

Before you perform this task, perform the “Configuring a BGP Routing Process” task.

Procedure

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 **interface** *type number*

Example:

Enters interface configuration mode.

Step 4 **vrf forwarding** *vrf-name*

Example:

```
Device(config-if)# vrf forwarding vpn1
```

Associates a VPN VRF instance with an interface or subinterface.

Step 5 **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]

Example:

```
Device(config-if)# ip address 192.168.3.1 255.255.255.0
```

Sets an IP address for an interface.

Step 6 **exit**

Example:

```
Device(config-if)# exit
```

Exits interface configuration mode and enters global configuration mode.

Step 7 **ip vrf** *vrf-name*

Example:

```
Device(config)# ip vrf vpn1
```

Configures a VRF routing table and enters VRF configuration mode.

- Use the *vrf-name* argument to specify a name to be assigned to the VRF.

Step 8 **rd** *route-distinguisher*

Example:

```
Device(config-vrf)# rd 45000:5
```

Creates routing and forwarding tables and specifies the default route distinguisher for a VPN.

- Use the *route-distinguisher* argument to add an 8-byte value to an IPv4 prefix to create a unique VPN IPv4 prefix.

Step 9 **route-target** {**import** | **export** | **both**} *route-target-ext-community*

Example:

```
Device(config-vrf)# route-target both 45000:100
```

Creates a route target extended community for a VRF.

- Use the **import** keyword to import routing information from the target VPN extended community.
- Use the **export** keyword to export routing information to the target VPN extended community.
- Use the **both** keyword to import both import and export routing information to the target VPN extended community.
- Use the *route-target-ext-community* argument to add the route target extended community attributes to the VRF's list of import, export, or both (import and export) route target extended communities.

Step 10 **exit**

Example:

```
Device(config-vrf)# exit
```

Exits VRF configuration mode and enters global configuration mode.

Step 11 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 12 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 vrf vpn1
```

Specifies the IPv4 address family and enters address family configuration mode.

- Use the **unicast** keyword to specify the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command.
- Use the **multicast** keyword to specify IPv4 multicast address prefixes.
- Use the **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.

Step 13 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 remote-as 50000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 14 **neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**]

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 maximum-prefix 10000 warning-only
```

Controls how many prefixes can be received from a neighbor.

- Use the *maximum* argument to specify the maximum number of prefixes allowed from the specified neighbor. The number of prefixes that can be configured is limited only by the available system resources on a device.
- Use the *threshold* argument to specify an integer representing a percentage of the maximum prefix limit at which the device starts to generate a warning message.
- Use the **warning-only** keyword to allow the device to generate a log message when the maximum prefix limit is exceeded, instead of terminating the peering session.

Step 15 **neighbor** *ip-address* **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 activate
```

Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local device.

Step 16 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

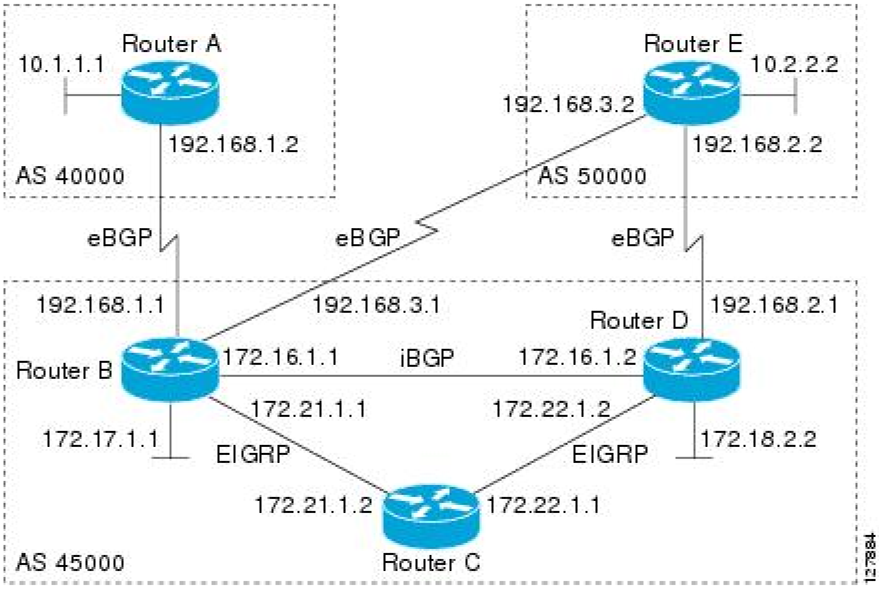
Use the **ping vrf** command to verify basic network connectivity between the BGP devices, and use the **show ip vrf** command to verify that the VRF instance has been created.

Customizing a BGP Peer

Perform this task to customize your BGP peers. Although many of the steps in this task are optional, this task demonstrates how the neighbor and address family configuration command relationships work. Using the example of the IPv4 multicast address family, neighbor address family-independent commands are configured before the IPv4 multicast address family is configured. Commands that are address family-dependent are then configured and the **exit address-family** command is shown. An optional step shows how to disable a neighbor.

The configuration in this task is done at Router B in the figure below and would need to be repeated with appropriate changes to the IP addresses, for example, at Router E to fully configure a BGP process between the two devices.

Figure 4: BGP Peer Topology



Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types, such as IPv6 prefixes.

Note When you configure an artificial default route with the **Neighbor** ip-address**default-originate** command, it obtains only the desired path attribute value changes if the route-map statement is added directly as an appendix.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **no bgp default ipv4-unicast**

Example:

```
Device(config-router)# no bgp default ipv4-unicast
```

Disables the IPv4 unicast address family for the BGP routing process.

Note Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.

Step 5 **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.3.2 remote-as 50000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 6 **neighbor** {*ip-address* | *peer-group-name*} **description** *text*

Example:

```
Device(config-router)# neighbor 192.168.3.2 description finance
```

(Optional) Associates a text description with the specified neighbor.

Step 7 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 multicast
```

Specifies the IPv4 address family and enters address family configuration mode.

- The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command.
- The **multicast** keyword specifies IPv4 multicast address prefixes.
- The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.

Step 8 **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]

Example:

```
Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0
```

(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table.

- For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.

Step 9 **neighbor** {*ip-address* | *peer-group-name*} **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 activate
```

Enables the exchange of information with a BGP neighbor.

Step 10 **neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds*

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 advertisement-interval 25
```

(Optional) Sets the minimum interval between the sending of BGP routing updates.

Step 11 **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]

Example:

```
Device(config-router-af)# neighbor 192.168.3.2 default-originate
```

(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.

Step 12 **exit-address-family**

Example:

```
Device(config-router-af)# exit-address-family
```

Exits address family configuration mode and enters router configuration mode.

Step 13 **neighbor** {*ip-address* | *peer-group-name*} **shutdown**

Example:

```
Device(config-router)# neighbor 192.168.3.2 shutdown
```

(Optional) Disables a BGP peer or peer group.

Note If you perform this step you will not be able to run either of the subsequent **show** command steps because you have disabled the neighbor.

Step 14 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 15 **show ip bgp ipv4 multicast** [*command*]

Example:

```
Device# show ip bgp ipv4 multicast
```

(Optional) Displays IPv4 multicast database-related information.

- Use the *command* argument to specify any multiprotocol BGP command that is supported. To see the supported commands, use the ? prompt on the CLI.

Step 16 **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received prefix-filter**]

Example:

```
Device# show ip bgp neighbors 192.168.3.2
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

Examples

The following sample output from the **show ip bgp ipv4 multicast** command shows BGP IPv4 multicast information for Router B in the figure above after this task has been configured on Router B and Router E. Note that the networks local to each device that were configured under IPv4 multicast address family appear in the output table.

```
BGP table version is 3, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.2.2.0/24      192.168.3.2              0             0 50000 i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
```

The following partial sample output from the **show ip bgp neighbors** command for neighbor 192.168.3.2 shows general BGP information and specific BGP IPv4 multicast address family information about the neighbor. The command was entered on Router B in the figure above after this task had been configured on Router B and Router E.

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Description: finance
  BGP version 4, remote router ID 10.2.2.99
  BGP state = Established, up for 01:48:27
  Last read 00:00:26, last write 00:00:26, hold time is 120, keepalive intervals
  Configured hold time is 120,keepalive interval is 70 seconds, Minimum holdtims
  Neighbor capabilities:
    Route refresh: advertised and received (old & new)
    Address family IPv4 Unicast: advertised
    Address family IPv4 Multicast: advertised and received
!
 For address family: IPv4 Multicast
  BGP table version 3, neighbor version 3/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
    Uses NEXT_HOP attribute for MBGP NLRIs
                                 Sent       Rcvd
  Prefix activity:               ----       ----
    Prefixes Current:               1          1 (Consumes 48 bytes)
    Prefixes Total:                 1          1
    Implicit Withdraw:              0          0
    Explicit Withdraw:              0          0
    Used as bestpath:             n/a          1
    Used as multipath:            n/a          0
                                   Outbound    Inbound
  Local Policy Denied Prefixes:    --------    -------
    Bestpath from this peer:              1        n/a
    Total:                                1          0
  Number of NLRIs in the update sent: max 0, min 0
  Minimum time between advertisement runs is 25 seconds
  Connections established 8; dropped 7
  Last reset 01:48:54, due to User reset
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 192.168.3.1, Local port: 13172
Foreign host: 192.168.3.2, Foreign port: 179
!
```

Removing BGP Configuration Commands Using a Redistribution

BGP CLI configuration can become quite complex even in smaller BGP networks. If you need to remove any CLI configuration, you must consider all the implications of removing the CLI. Analyze the current running configuration to determine the current BGP neighbor relationships, any address family considerations, and even other routing protocols that are configured. Many BGP CLI commands affect other parts of the CLI configuration.

Perform this task to remove all the BGP configuration commands used in a redistribution of BGP routes into EIGRP. A route map can be used to match and set parameters or to filter the redistributed routes to ensure that routing loops are not created when these routes are subsequently advertised by EIGRP. When removing BGP configuration commands you must remember to remove or disable all the related commands. In this example, if the **route-map** command is omitted, then the redistribution will still occur and possibly with unexpected results as the route map filtering has been removed. Omitting just the **redistribute** command would mean that the route map is not applied, but it would leave unused commands in the running configuration.

For more details on BGP CLI removal, see the “BGP CLI Removal Considerations” concept in the “Cisco BGP Overview” module.

To view the redistribution configuration before and after the CLI removal, see the “Examples: Removing BGP Configuration Commands Using a Redistribution Example” section.

Procedure

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 **no route-map** *map-name*

Example:

```
Device(config)# no route-map bgp-to-eigrp
```

Removes a route map from the running configuration.

- In this example, a route map named bgp-to-eigrp is removed from the configuration.

Step 4 **router eigrp** *autonomous-system-number*

Example:

```
Device(config)# router eigrp 100
```

Enters router configuration mode for the specified routing process.

Step 5 **no redistribute** *protocol* [*as-number*]

Example:

```
Device(config-router)# no redistribute bgp 45000
```

Disables the redistribution of routes from one routing domain into another routing domain.

- In this example, the configuration of the redistribution of BGP routes into the EIGRP routing process is removed from the running configuration.

Note If a route map was included in the original **redistribute** command configuration, remember to remove the **route-map** command configuration as in Step 3 in this example task.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 6 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Step 7 **show running-config**

Example:

```
Device# show running-config
```

(Optional) Displays the current running configuration on the router.

- Use this command to verify that the **redistribute** and **route-map** commands are removed from the router configuration.

Monitoring and Maintaining Basic BGP

The tasks in this section are concerned with the resetting and display of information about basic BGP processes and peer relationships. Once you have defined two devices to be BGP neighbors, they will form a BGP connection and exchange routing information. If you subsequently change a BGP filter, weight, distance, version, or timer, or make a similar configuration change, you may have to reset BGP connections for the configuration change to take effect.

Configuring Inbound Soft Reconfiguration When Route Refresh Capability Is Missing

Perform this task to configure inbound soft reconfiguration using the **bgp soft-reconfig-backup** command for BGP peers that do not support the route refresh capability. BGP peers that support the route refresh capability are unaffected by the configuration of this command. Note that the memory requirements for storing the inbound update information can become quite large.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **bgp log-neighbor-changes**

Example:

```
Device(config-router)# bgp log-neighbor-changes
```

Enables logging of BGP neighbor resets.

Step 5 **bgp soft-reconfig-backup**

Example:

```
Device(config-router)# bgp soft-reconfig-backup
```

Configures a BGP speaker to perform inbound soft reconfiguration for peers that do not support the route refresh capability.

- This command is used to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.

Step 6 **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 40000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 7 **neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]

Example:

```
Device(config-router)# neighbor 192.168.1.2 soft-reconfiguration inbound
```

Configures the Cisco software to start storing updates.

- All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is done later, the stored information will be used to generate a new set of inbound updates.

Step 8 **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}

Example:

```
Device(config-router)# neighbor 192.168.1.2 route-map LOCAL in
```

Applies a route map to incoming or outgoing routes.

- In this example, the route map named LOCAL will be applied to incoming routes.

Step 9 Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.

—

Step 10 **exit**

Example:

```
Device(config-router)# exit
```

Exits router configuration mode and enters global configuration mode.

Step 11 **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]

Example:

```
Device(config)# route-map LOCAL permit 10
```

Configures a route map and enters route-map configuration mode.

- In this example, a route map named LOCAL is created.

Step 12 **set ip next-hop** *ip-address*

Example:

```
Device(config-route-map)# set ip next-hop 192.168.1.144
```

Specifies where output packets that pass a match clause of a route map for policy routing.

- In this example, the ip address is set to 192.168.1.144.

Step 13 **end**

Example:

```
Device(config-route-map)# end
```

Exits route-map configuration mode and enters privileged EXEC mode.

Step 14 **show ip bgp neighbors** [*neighbor-address*]

Example:

```
Device# show ip bgp neighbors 192.168.1.2
```

(Optional) Displays information about the TCP and BGP connections to neighbors.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 15 **show ip bgp** [*network*] [*network-mask*]

Example:

```
Device# show ip bgp
```

(Optional) Displays the entries in the BGP routing table.

Note Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following partial output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.2.1. This peer supports route refresh.

```
BGP neighbor is 192.168.1.2,  remote AS 40000, external link
 Neighbor capabilities:
   Route refresh: advertised and received(new)
```

The following partial output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.3.2. This peer does not support route refresh so the soft-reconfig inbound paths for BGP peer 192.168.3.2 will be stored because there is no other way to update any inbound policy updates.

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Neighbor capabilities:
   Route refresh: advertised
```

The following sample output from the **show ip bgp** command shows the entry for the network 172.17.1.0. Both BGP peers are advertising 172.17.1.0/24, but only the received-only path is stored for 192.168.3.2.

```
BGP routing table entry for 172.17.1.0/24, version 11
Paths: (3 available, best #3, table Default-IP-Routing-Table, RIB-failure(4))
Flag: 0x820
```

```
Advertised to update-groups:
   1
50000
  192.168.3.2 from 192.168.3.2 (172.17.1.0)
    Origin incomplete, metric 0, localpref 200, valid, external
50000, (received-only)
  192.168.3.2 from 192.168.3.2 (172.17.1.0)
    Origin incomplete, metric 0, localpref 100, valid, external
40000
  192.168.1.2 from 192.168.1.2 (172.16.1.0)
    Origin incomplete, metric 0, localpref 200, valid, external, best
```

Resetting and Displaying Basic BGP Information

Perform this task to reset and display information about basic BGP processes and peer relationships.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} [**soft** [**in** | **out**]]

Example:

```
Device# clear ip bgp *
```

Clears and resets BGP neighbor sessions:

- In the example provided, all BGP neighbor sessions are cleared and reset.

Step 3 **show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes** *mask-length*]

Example:

```
Device# show ip bgp 10.1.1.0 255.255.255.0
```

Displays all the entries in the BGP routing table:

- In the example provided, the BGP routing table information for the 10.1.1.0 network is displayed.

Step 4 **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*]

Example:

```
Device# show ip bgp neighbors 192.168.3.2 advertised-routes
```

Displays information about the TCP and BGP connections to neighbors.

- In the example provided, the routes advertised from the device to BGP neighbor 192.168.3.2 on another device are displayed.

Step 5 **show ip bgp paths**

Example:

```
Device# show ip bgp paths
```

Displays information about all the BGP paths in the database.

Step 6 **show ip bgp summary**

Example:

```
Device# show ip bgp summary
```

Displays information about the status of all BGP connections.

Aggregating Route Prefixes Using BGP

BGP peers exchange information about local networks, but this can quickly lead to large BGP routing tables. CIDR enables the creation of aggregate routes (or *supernets*) to minimize the size of routing tables. Smaller BGP routing tables can reduce the convergence time of the network and improve network performance. Aggregated routes can be configured and advertised using BGP. Some aggregations advertise only summary routes and other methods of aggregating routes allow more specific routes to be forwarded. Aggregation applies only to routes that exist in the BGP routing table. An aggregated route is forwarded if at least one more specific route of the aggregation exists in the BGP routing table. Perform one of the following tasks to aggregate routes within BGP:

Redistributing a Static Aggregate Route into BGP

Use this task to redistribute a static aggregate route into BPG. A static aggregate route is configured and then redistributed into the BGP routing table. The static route must be configured to point to interface null 0 and the prefix should be a superset of known BGP routes. When a device receives a BGP packet, it will use the more specific BGP routes. If the route is not found in the BGP routing table, then the packet will be forwarded to null 0 and discarded.

Procedure

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 route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]

Example:

```
Device(config)# ip route 172.17.0.0 255.0.0.0 null 0
```

Creates a static route.

Step 4 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 5 **redistribute static**

Example:

```
Device(config-router)# redistribute static
```

Redistributes routes into the BGP routing table.

Step 6 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and returns to privileged EXEC mode.

Configuring Conditional Aggregate Routes Using BGP

Use this task to create an aggregate route entry in the BGP routing table when at least one specific route falls into the specified range. The aggregate route is advertised as originating from your autonomous system. For more information, see the “BGP Route Aggregation Generating AS_SET Information” section.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **aggregate-address** *address mask* [**as-set**]

Example:

```
Device(config-router)# aggregate-address 172.17.0.0 255.0.0.0 as-set
```

Creates an aggregate entry in a BGP routing table.

- A specified route must exist in the BGP table.
- Use the **aggregate-address** command with no keywords to create an aggregate entry if any more-specific BGP routes are available that fall in the specified range.
- Use the **as-set** keyword to specify that the path advertised for this route is an AS_SET. Do not use the **as-set** keyword when aggregating many paths because this route is withdrawn and updated every time the reachability information for the aggregated route changes.

Note Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 5 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Suppressing and Unsuppressing the Advertisement of Aggregated Routes Using BGP

Use this task to create an aggregate route, suppress the advertisement of routes using BGP, and subsequently unsuppress the advertisement of routes. Routes that are suppressed are not advertised to any neighbors, but it is possible to unsuppress routes that were previously suppressed to specific neighbors.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 40000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5 Do one of the following:

- **aggregate-address** *address mask* [**summary-only**]
- **aggregate-address** *address mask* [**suppress-map** *map-name*]

Example:

```
Device(config-router)# aggregate-address 172.17.0.0 255.0.0.0 summary-only
```

Example:

```
Device(config-router)# aggregate-address 172.17.0.0 255.0.0.0 suppress-map map1
```

Creates an aggregate route.

- Use the optional **summary-only** keyword to create the aggregate route (for example, 10.*.*.*) and also suppresses advertisements of more-specific routes to all neighbors.

- Use the optional **suppress-map** keyword to create the aggregate route but suppress advertisement of specified routes. Routes that are suppressed are not advertised to any neighbors. You can use the **match** clauses of route maps to selectively suppress some more-specific routes of the aggregate and leave others unsuppressed. IP access lists and autonomous system path access lists **match** clauses are supported.

Note Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 6 **neighbor** {*ip-address* | *peer-group-name*} **unsuppress-map** *map-name*

Example:

```
Device(config-router)# neighbor 192.168.1.2 unsuppress map1
```

(Optional) Selectively advertises routes previously suppressed by the **aggregate-address** command.

- In this example, the routes previously suppressed in Step 5 are advertised to neighbor 192.168.1.2.

Step 7 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Conditionally Advertising BGP Routes

Perform this task to conditionally advertise selected BGP routes. The routes or prefixes that will be conditionally advertised are defined in two route maps: an advertise map and either an exist map or nonexist map. The route map associated with the exist map or nonexist map specifies the prefix that the BGP speaker will track. The route map associated with the advertise map specifies the prefix that will be advertised to the specified neighbor when the condition is met.

- If a prefix is found to be present in the exist map by the BGP speaker, the prefix specified by the advertise map is advertised.
- If a prefix is found not to be present in the nonexist map by the BGP speaker, the prefix specified by the advertise map is advertised.

If the condition is not met, the route is withdrawn and conditional advertisement does not occur. All routes that may be dynamically advertised or not advertised must exist in the BGP routing table in order for conditional advertisement to occur. These routes are referenced from an access list or an IP prefix list.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 40000
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5 **neighbor** *ip-address* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*}

Example:

```
Device(config-router)# neighbor 192.168.1.2 advertise-map map1 exist-map map2
```

Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

- In this example, the prefix (172.17.0.0) matching the ACL in the advertise map (the route map named map1) will be advertised to the neighbor only when a prefix (192.168.50.0) matching the ACL in exist map (the route-map named map2) is in the local BGP table.

Step 6 **exit**

Example:

```
Device(config-router)# exit
```

Exits router configuration mode and enters global configuration mode.

Step 7 **route-map** *map-tag* [**permit** | **deny**] [**sequence-number**]

Example:

```
Device(config)# route-map map1 permit 10
```

Configures a route map and enters route map configuration mode.

- In this example, a route map named map1 is created.

Step 8 **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}

Example:

```
Device(config-route-map)# match ip address 1
```

Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

- In this example, the route map is configured to match a prefix permitted by access list 1.

Step 9 **exit**

Example:

```
Device(config-route-map)# exit
```

Exits route map configuration mode and enters global configuration mode.

Step 10 **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]

Example:

```
Device(config)# route-map map2 permit 10
```

Configures a route map and enters route map configuration mode.

- In this example, a route map named map2 is created.

Step 11 **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}

Example:

```
Device(config-route-map)# match ip address 2
```

Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

- In this example, the route map is configured to match a prefix permitted by access list 2.

Step 12 **exit**

Example:

```
Device(config-route-map)# exit
```

Exits route map configuration mode and enters global configuration mode.

Step 13 **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]

Example:

```
Device(config)# access-list 1 permit 172.17.0.0
```

Configures a standard access list.

- In this example, access list 1 permits advertising of the 172.17.0.0 prefix, depending on other conditions set by the **neighbor advertise-map** command.

Step 14 **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]

Example:

```
Device(config)# access-list 2 permit 192.168.50.0
```

Configures a standard access list.

- In this example, access list 2 permits the 192.168.50.0 to be the prefix of the exist-map.

Step 15 **exit**

Example:

```
Device(config)# exit
```

Exits global configuration mode and returns to privileged EXEC mode.

Originating BGP Routes

Route aggregation is useful to minimize the size of the BGP table, but there are situations when you want to add more specific prefixes to the BGP table. Route aggregation can hide more specific routes. Using the **network** command as shown in the “Configuring a BGP Routing Process” section originates routes, and the following optional tasks originate BGP routes for the BGP table for different situations.

Advertising a Default Route Using BGP

Perform this task to advertise a default route to BGP peers. The default route is locally originated. A default route can be useful to simplify configuration or to prevent the device from using too many system resources. If the device is peered with an Internet service provider (ISP), the ISP will carry full routing tables, so configuring a default route into the ISP network saves resources at the local device.

Procedure

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 prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network* / *length* | **permit** *network* / *length*} [**ge** *ge-value*] [**le** *le-value*]

Example:

```
Device(config)# ip prefix-list DEFAULT permit 10.1.1.0/24
```

Configures an IP prefix list.

- In this example, prefix list DEFAULT permits advertising of the 10.1.1.0/24. prefix depending on a match set by the **match ip address** command.

Step 4 **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]

Example:

```
Device(config)# route-map ROUTE
```

Configures a route map and enters route map configuration mode.

- In this example, a route map named ROUTE is created.

Step 5 **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}

Example:

```
Device(config-route-map)# match ip address prefix-list DEFAULT
```

Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

- In this example, the route map is configured to match a prefix permitted by prefix list DEFAULT.

Step 6 **exit**

Example:

```
Device(config-route-map)# exit
```

Exits route map configuration mode and enters global configuration mode.

Step 7 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Enters router configuration mode for the specified routing process.

Step 8 **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]

Example:

```
Device(config-router)# neighbor 192.168.3.2 default-originate
```

(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.

Step 9 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and enters privileged EXEC mode.

Originating BGP Routes Using Backdoor Routes

Use this task to indicate to border devices which networks are reachable using a backdoor route. A backdoor network is treated the same as a local network, except that it is not advertised. For more information, see the BGP Backdoor Routes section.

Before you begin

This task assumes that the IGP (EIGRP, in this example) is already configured for the BGP peers. The configuration is done at Router B in the figure 1 in the “BGP Backdoor Routes” section, and the BGP peer is Router D.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 45000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 172.22.1.2 remote-as 45000
```

Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device.

- In this example, the peer is an internal peer as the autonomous system number specified for the peer is the same number specified in Step 3.

Step 5 **network** *ip-address* **backdoor**

Example:

```
Device(config-router)# network 172.21.1.0 backdoor
```

Indicates a network that is reachable through a backdoor route.

Step 6 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and returns to privileged EXEC mode.

Configuring a BGP Peer Group

This task explains how to configure a BGP peer group. Often, in a BGP speaker, many neighbors are configured with the same update policies (that is, the same outbound route maps, distribute lists, filter lists, update source, and so on). Neighbors with the same update policies can be grouped into peer groups to simplify configuration and, more importantly, to make updating more efficient. When you have many peers, this approach is highly recommended.

The three steps to configure a BGP peer group, described in the following task, are as follows:

- Creating the peer group
- Assigning options to the peer group
- Making neighbors members of the peer group

You can disable a BGP peer or peer group without removing all the configuration information using the **neighbor shutdown** router configuration command.

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 40000
```

Enters router configuration mode for the specified routing process.

Step 4 **neighbor** *peer-group-name* **peer-group**

Example:

```
Device(config-router)# neighbor fingroup peer-group
```

Creates a BGP peer group.

Step 5 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.1 remote-as 45000
```

Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device.

Step 6 **neighbor** *ip-address* **peer-group** *peer-group-name*

Example:

```
Device(config-router)# neighbor 192.168.1.1 peer-group fingroup
```

Assigns the IP address of a BGP neighbor to a peer group.

Step 7 **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

Example:

```
Device(config-router)# address-family ipv4 multicast
```

Specifies the IPv4 address family and enters address family configuration mode.

- The **unicast** keyword specifies the IPv4 unicast address family. This is the default.
- The **multicast** keyword specifies that IPv4 multicast address prefixes will be exchanged.
- The **vrf** keyword and *vrf-name* argument specify that IPv4 VRF instance information will be exchanged.

Step 8 **neighbor** *peer-group-name* **activate**

Example:

```
Device(config-router-af)# neighbor fingroup activate
```

Enables the neighbor to exchange prefixes for the IPv4 address family with the local device.

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only unicast address prefixes. To allow BGP to exchange other address prefix types, such as multicast that is configured in this example, neighbors must also be activated using the **neighbor activate** command.

Step 9 **neighbor** *ip-address* **peer-group** *peer-group-name*

Example:

```
Device(config-router-af)# neighbor 192.168.1.1 peer-group fingroup
```

Assigns the IP address of a BGP neighbor to a peer group.

Step 10 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP 4

Example: Configuring a BGP Process and Customizing Peers

The following example shows the configuration for Router B in the figure 4 above (in the “Customizing a BGP Peer” section) with a BGP process configured with two neighbor peers (at Router A and at Router E) in separate autonomous systems. IPv4 unicast routes are exchanged with both peers and IPv4 multicast routes are exchanged with the BGP peer at Router E.

Router B

```
router bgp 45000
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
 !
 address-family ipv4 multicast
  neighbor 192.168.3.2 activate
  neighbor 192.168.3.2 advertisement-interval 25
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

Examples: Removing BGP Configuration Commands Using a Redistribution Example

The following examples show first the CLI configuration to enable the redistribution of BGP routes into EIGRP using a route map and then the CLI configuration to remove the redistribution and route map. Some BGP configuration commands can affect other CLI commands and this example demonstrates how the removal of one command affects another command.

In the first configuration example, a route map is configured to match and set autonomous system numbers. BGP neighbors in three different autonomous systems are configured and activated. An EIGRP routing process is started, and the redistribution of BGP routes into EIGRP using the route map is configured.

CLI to Enable BGP Route Redistribution Into EIGRP

```
route-map bgp-to-eigrp permit 10
 match tag 50000
 set tag 65000
 exit
router bgp 45000
 bgp log-neighbor-changes
 address-family ipv4
  neighbor 172.16.1.2 remote-as 45000
  neighbor 172.21.1.2 remote-as 45000
  neighbor 192.168.1.2 remote-as 40000
  neighbor 192.168.3.2 remote-as 50000
  neighbor 172.16.1.2 activate
  neighbor 172.21.1.2 activate
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
 exit
router eigrp 100
 redistribute bgp 45000 metric 10000 100 255 1 1500 route-map bgp-to-eigrp
 no auto-summary
 exit
```

In the second configuration example, both the **route-map** command and the **redistribute** command are disabled. If only the route-map command is removed, it does not automatically disable the redistribution. The redistribution will now occur without any matching or filtering. To remove the redistribution configuration, the **redistribute** command must also be disabled.

CLI to Remove BGP Route Redistribution Into EIGRP

```
configure terminal
 no route-map bgp-to-eigrp
 router eigrp 100
  no redistribute bgp 45000
  end
```

Examples: BGP Soft Reset

The following examples show two ways to reset the connection for BGP peer 192.168.1.1.

Example: Dynamic Inbound Soft Reset

The following example shows the command used to initiate a dynamic soft reconfiguration in the BGP peer 192.168.1.1. This command requires that the peer support the route refresh capability.

```
clear ip bgp 192.168.1.1 soft in
```

Example: Inbound Soft Reset Using Stored Information

The following example shows how to enable inbound soft reconfiguration for the neighbor 192.168.1.1. All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is performed later, the stored information will be used to generate a new set of inbound updates.

```
router bgp 100
 neighbor 192.168.1.1 remote-as 200
 neighbor 192.168.1.1 soft-reconfiguration inbound
```

The following example clears the session with the neighbor 192.168.1.1:

```
clear ip bgp 192.168.1.1 soft in
```

Example: Resetting and Displaying Basic BGP Information

The following example shows how to reset and display basic BGP information.

The **clear ip bgp *** command clears and resets all the BGP neighbor sessions. Specific neighbors or all peers in an autonomous system can be cleared by using the *neighbor-address* and *autonomous-system-number* arguments. If no argument is specified, this command will clear and reset all BGP neighbor sessions.

Note The **clear ip bgp *** command also clears all the internal BGP structures, which makes it useful as a troubleshooting tool.

```
Device# clear ip bgp *
```

The **show ip bgp** command is used to display all the entries in the BGP routing table. The following example displays BGP routing table information for the 10.1.1.0 network:

```
Device# show ip bgp 10.1.1.0 255.255.255.0

BGP routing table entry for 10.1.1.0/24, version 2
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  40000
    192.168.1.2 from 192.168.1.2 (10.1.1.99)
      Origin IGP, metric 0, localpref 100, valid, external, best
```

The **show ip bgp neighbors** command is used to display information about the TCP and BGP connections to neighbors. The following example displays the routes that were advertised from Router B in the figure above (in the “Configuring a BGP Peer for the IPv4 VRF Address Family” section) to its BGP neighbor 192.168.3.2 on Router E:

```
Device# show ip bgp neighbors 192.168.3.2 advertised-routes

BGP table version is 3, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 40000 i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
Total number of prefixes 2
```

The **show ip bgp paths** command is used to display all the BGP paths in the database. The following example displays BGP path information for Router B in the figure above (in the “Customizing a BGP Peer” section):

```
Device# show ip bgp paths

Address    Hash Refcount Metric Path
0x2FB5DB0     0        5      0 i
0x2FB5C90     1        4      0 i
0x2FB5C00  1361        2      0 50000 i
0x2FB5D20  2625        2      0 40000 i
```

The **show ip bgp summary** command is used to display the status of all BGP connections. The following example displays BGP routing table information for Router B in the figure above (in the “Customizing a BGP Peer” section):

```
Device# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 45000
BGP table version is 3, main routing table version 3
2 network entries using 234 bytes of memory
2 path entries using 104 bytes of memory
4/2 BGP path/bestpath attribute entries using 496 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 882 total bytes of memory
BGP activity 14/10 prefixes, 16/12 paths, scan interval 60 secs
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.168.1.2     4 40000     667     672        3    0    0 00:03:49        1
192.168.3.2     4 50000     468     467        0    0    0 00:03:49 (NoNeg)
```

Examples: Aggregating Prefixes Using BGP

The following examples show how you can use aggregate routes in BGP either by redistributing an aggregate route into BGP or by using the BGP conditional aggregation routing feature.

In the following example, the **redistribute static** router configuration command is used to redistribute aggregate route 10.0.0.0:

```
ip route 10.1.0.0 255.0.0.0 null 0
!
router bgp 100
 redistribute static
```

The following configuration shows how to create an aggregate entry in the BGP routing table when at least one specific route falls into the specified range. The aggregate route will be advertised as coming from your autonomous system and has the atomic aggregate attribute set to show that information might be missing. (By default, atomic aggregate is set unless you use the **as-set** keyword in the **aggregate-address** router configuration command.)

```
router bgp 100
 aggregate-address 10.1.0.0 255.0.0.0
```

The following example shows how to create an aggregate entry using the same rules as in the previous example, but the path advertised for this route will be an AS_SET consisting of all elements contained in all paths that are being summarized:

```
router bgp 100
 aggregate-address 10.1.0.0 255.0.0.0 as-set
```

The following example shows how to create the aggregate route for 10.0.0.0 and also suppress advertisements of more specific routes to all neighbors:

```
router bgp 100
 aggregate-address 10.1.0.0 255.0.0.0 summary-only
```

The following example configures BGP to not advertise inactive routes:

```
Device(config)# router bgp 50000
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# bgp suppress-inactive
Device(config-router-af)# end
```

The following example configures a maximum route limit in the VRF named RED and configures BGP to not advertise inactive routes through the VRF named RED:

```
Device(config)# ip vrf RED
Device(config-vrf)# rd 50000:10
Device(config-vrf)# maximum routes 1000 10
Device(config-vrf)# exit
Device(config)# router bgp 50000
Device(config-router)# address-family ipv4 vrf RED
Device(config-router-af)# bgp suppress-inactive
Device(config-router-af)# end
```

Example: Configuring a BGP Peer Group

The following example shows how to use an address family to configure a peer group so that all members of the peer group are both unicast- and multicast-capable:

```
router bgp 45000
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 address-family ipv4 unicast
  neighbor mygroup peer-group
  neighbor 192.168.1.2 peer-group mygroup
  neighbor 192.168.3.2 peer-group mygroup
router bgp 45000
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 address-family ipv4 multicast
  neighbor mygroup peer-group
  neighbor 192.168.1.2 peer-group mygroup
  neighbor 192.168.3.2 peer-group mygroup
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference

Standards and RFCs

Standard/RFC	Title
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1773	*Experience with the BGP Protocol*
RFC 1774	*BGP-4 Protocol Analysis*
RFC 1930	*Guidelines for Creation, Selection, and Registration on an Autonomous System (AS)*
RFC 2519	*A Framework for Inter-Domain Route Aggregation*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 2918	*Route Refresh Capability for BGP-4*
RFC 3392	*Capabilities Advertisement with BGP-4*

Standard/RFC	Title
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*

MIBs

MIB	MIBs Link
CISCO-BGP4-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

CHAPTER 3

BGP PIC (Prefix Independent Convergence) Edge for IP and MPLS-VPN

The BGP PIC (Prefix Independent Convergence) Edge for IP and MPLS-VPN feature improves BGP convergence after a network failure. This convergence is applicable to both core and edge failures and can be used in both IP and MPLS networks. The BGP PIC Edge for IP and MPLS-VPN feature creates and stores a backup or alternate path in the routing information base (RIB), forwarding information base (FIB), and Cisco Express Forwarding. When a failure is detected, the backup or alternate path immediately takes over, thus enabling fast failover.

Note In this document, the BGP PIC Edge for IP and MPLS-VPN feature is called by the short name BGP PIC.

Prerequisites for BGP PIC

- Ensure that the Border Gateway Protocol (BGP) and the IP or Multiprotocol Label Switching (MPLS) network is up and running with the customer site connected to the provider site by more than one path (multihomed).
- Ensure that the backup/alternate path has a unique next hop that is not the same as the next hop of the best path.
- Enable the Bidirectional Forwarding Detection (BFD) protocol to quickly detect link failures of directly connected neighbors.
- Ensure the following output is displayed in the operational field for cef convergence -speed on the ASR 900 RSP3 Module:

```
RSP3#sh cef table
Global information:
Output chain build favors:
platform: not configured
CLI: not configured
operational: convergence-speed
```

Restrictions for BGP PIC

- The uLoop during cutover may delay the label switched paths propagation and eventually impact BGP PIC edge Convergence. To avoid uLoop, you can configure MPLS-TE tunnels from local PE to the RRs.
- If multiple features are configured with their respective scales, then it affects the BGP PIC edge Convergence.
- With BGP Multipath, the BGP Prefix-Independent Convergence (PIC) feature is already supported.
- In MPLS VPNs, the BGP PIC feature is not supported with MPLS VPN Inter-Autonomous Systems Option B.
- The BGP PIC feature supports prefixes only for IPv4, IPv6, VPNv4, and VPNv6 address families.
- The BGP PIC feature cannot be configured with Multicast or L2VPN Virtual Routing and Forwarding (VRF) address families.
- If the route reflector is only in the control plane, then you do not need BGP PIC, because BGP PIC addresses data plane convergence.
- When two PE routers become each other’s backup/alternate path to a CE router, traffic might loop if the CE router fails. Neither router will reach the CE router, and traffic will continue to be forwarded between the PE routers until the time-to-live (TTL) timer expires.
- The BGP PIC feature solves the traffic forwarding only for a single network failure at both the edge and the core.
- The BGP PIC feature does not work with the BGP Best External feature. If you try to configure the BGP PIC feature after configuring the BGP Best External feature, you receive an error.
- BGP PIC over bridge domain interface (BDI) core interfaces can have only one Ethernet Flow Point (EFP) associated with each of the BDI interfaces.
- The maximum number of bridge domain interfaces (BDI) that can act as protected or protecting interfaces via BGP PIC is 24.
- BGP PIC Core is enabled only if a minimum of 15 BGP prefixes are received from the peer.
- BGP PIC edge works only when BGP PIC core is enabled.
- Each of the 2 BGP peers must send the same set of 15 BGP prefixes to the DUT.
- BGP PIC edge provides sub-second convergence for Global prefixes, EoMPLS over BGP LU, VPLS, and 6PE.
- BGP PIC edge provides sub-second convergence for VPNv4/VPNv6 prefixes with a maximum of 4000 (including global). Sub-second convergence is not supported for BGP PIC Edge for prefixes beyond 4000.

- BGP PIC is only supported for Ethernet pseudowires and VPLS. Starting Cisco IOS XE Release 3.17, BGP PIC is supported for TDM pseudowires on the ASR 900 RSP1 module. For information on configuring pseudowires and optical interfaces, see Cisco ASR 900 Router Series Configuration Guide.
- If PIC Core is disabled and BGP send-label (3107) is configured, ECMP is not supported in PE node on the Cisco ASR 900 RSP3 Module.
- Starting with Cisco IOS XE Everest 16.6.1 release, **bgp mpls-local-label** command must be enabled for BGP PIC edge with EoMPLS over BGP labeled unicast (RFC3107) configuration.
- Starting with Cisco IOS XE Release 16.9.1, BGP PIC Edge is supported over TE-FRR with Next Hops.
- BGP PIC edge is supported only on dual-rate ports. It is not supported on the AMS ports.
- Targeted LDP (MPLS IP over tunnels) is not supported.
- VPLS over MPLS TE and MPLS TE FRR is not supported.
- Tunnel configuration under two simultaneous IGPs is not supported.
- Tunnel statistics is not supported, if one of the labels (Primary/Backup) is an implicit-null. For the tunnel statistics to work as expected:
  - Always configure tunnel which is at least one hop away and not the immediate next-hop.
  - In case of TE-FRR, the backup tunnel should end at least one hop before the primary tunnel's destination and not on the same hop as that of the primary tunnel's destination. However, for ping packets, statistics will increment.
- Sub second convergence is not guaranteed when more than 100 primary and backup tunnels are terminating on the same node.
- 500 MPLS TE Headend Tunnels are supported.
- Primary tunnels configured for link or node protection cannot go over port channel interfaces.
- P2MP TE Tunnels are not supported.
- Multicast over PtoP Tunnel (MPLS TE) is not supported.
- Inter AS TE tunnels are not supported.
- MPLS TE path protection is not supported.
- MPLS TE auto route destination not supported for inter-area tunnels.
- Use the following commands to avoid high convergence issues. Actual values for *delay installation* and *delay cleanup* can be configured.
  - **mpls traffic-eng reoptimize timers delay installation** *delay installation*
  - **mpls traffic-eng reoptimize timers delay cleanup** *delay cleanup*
- MPLS TE tunnel statisticss are supported only for tunnels with real label.
- MPLS TE explicit-null is not supported.

About BGP PIC

In the following sections, we describe the BGP PIC feature in details, how to detect a failure, a scenario and how to configure it.

Benefits

- An extra path for failover allows faster restoration of connectivity when a primary path is invalid or withdrawn.
- Reduction of traffic loss.
- Constant convergence time so that the switching time is the same for all prefixes.
- Starting with Cisco IOS XE Release 3.10S, both the Labeled Border Gateway Protocol (BGP) Prefix-Independent Convergence (PIC) feature and the Loop-Free Alternate (LFA) Fast Reroute (FRR) feature can be configured together on the router.

BGP Convergence

Under normal circumstances, BGP can take several seconds to a few minutes to converge after a change in the network. At a high level, BGP goes through the steps of the following process:

1. BGP learns of failures through either Interior Gateway Protocol (IGP) or BFD events or interface events.
2. BGP withdraws the routes from the routing information base (RIB), and the RIB withdraws the routes from the forwarding information base (FIB) and distributed FIB (dFIB). This process clears the data path for the affected prefixes.
3. BGP sends withdrawn messages to its neighbors.
4. BGP calculates the next best path to the affected prefixes.
5. BGP inserts the next best path for affected prefixes into the RIB, and the RIB installs them in the FIB and dFIB.

This process may take from few seconds to a few minutes to complete. It depends on, the latency of the network, the convergence time across the network, and the local load on the devices. The data plane converges only after the control plane converges.

Improve Convergence

The BGP PIC functionality is achieved by an extra functionality in the BGP, RIB, Cisco Express Forwarding, and MPLS.

- BGP Functionality

BGP PIC affects prefixes under IPv4 and VPNv4 address families. For those prefixes, BGP calculates an extra second best path, along with the primary best path. (The second best path is called the backup or alternate path.) BGP installs the best and backup or alternate paths for the affected prefixes into the BGP RIB. The backup or alternate path provides a fast reroute mechanism to counter a singular network failure. BGP also includes the alternate or backup path in its application programming interface (API) to the IP RIB.

- RIB Functionality

For BGP PIC, RIB installs an alternate path per route if one is available. If the RIB selects a BGP route containing a backup or alternate path, it installs the backup or alternate path with the best path. The RIB also includes the alternate path in its API with the FIB.

- Cisco Express Forwarding Functionality

With BGP PIC, Cisco Express Forwarding stores an alternate path per prefix. When the primary path goes down, Cisco Express Forwarding searches for the backup or alternate path in a prefix-independent manner. Cisco Express Forwarding also listens to BFD events to rapidly detect local failures.

- MPLS Functionality

MPLS Forwarding is similar to Cisco Express Forwarding in that it stores alternate paths and switches to an alternate path if the primary path goes down.

When the BGP PIC feature is enabled, BGP calculates a backup or alternate path per prefix and installs it into BGP RIB, IP RIB, and FIB. This improves convergence after a network failure. There are two types of network failures that the BGP PIC feature detects:

- Core node or link failure (internal Border Gateway Protocol [iBGP] node failure): If a PE node or link fails, then the failure is detected through IGP convergence. IGP conveys the failure through the RIB to the FIB.
- Local link or immediate neighbor node failure (external Border Gateway Protocol [eBGP] node or link failure): To detect a local link failure or eBGP single-hop peer node failure in less than a second, you must enable BFD. Cisco Express Forwarding looks for BFD events to detect a failure of an eBGP single-hop peer.

Convergence in the Data Plane

Upon detecting a failure, Cisco Express Forwarding detects the alternate next hop for all prefixes that are affected by the failure. The data plane convergence is achieved in subseconds depending on whether the BGP PIC implementation exists in the software or hardware.

Convergence in the Control Plane

Upon detecting a failure, BGP learns about the failure through IGP convergence or BFD events and sends withdrawn messages for the prefixes, recalculating the best and backup or alternate paths, and advertising the next best path across the network.

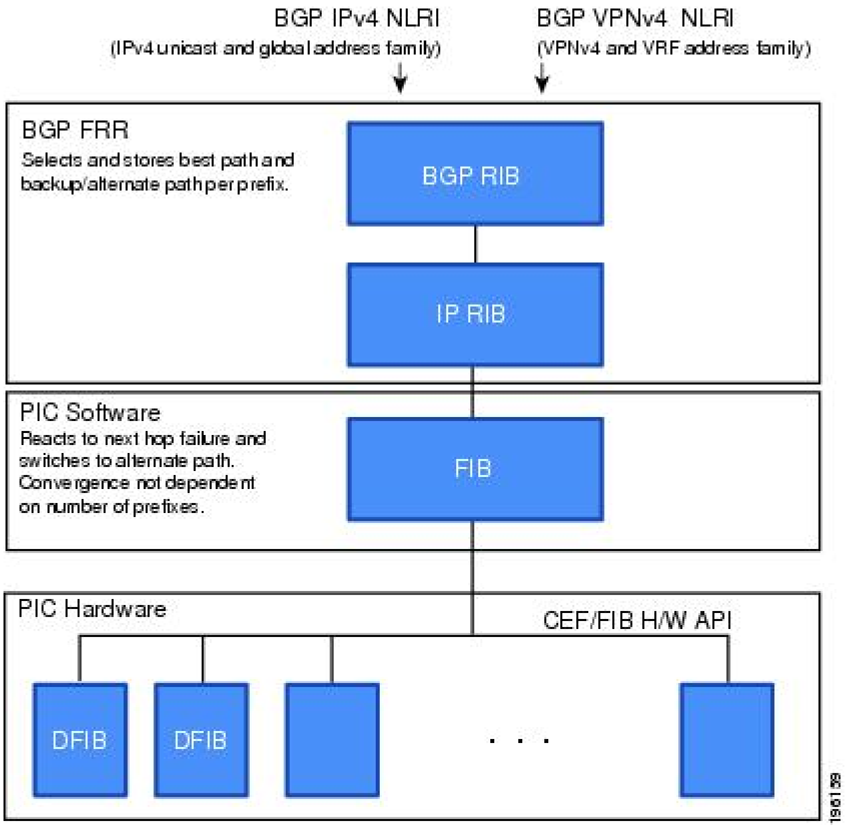
BGP Fast Reroute

BGP Fast Reroute (FRR) provides a best path and a backup or alternate path in BGP, RIB, and Cisco Express Forwarding. BGP FRR provides a fast reroute mechanism into the RIB and Cisco Express Forwarding (CEF) on the backup BGP next hop to reach a destination when the current best path is not available.

BGP FRR precomputes a second best path in BGP and gives it to the RIB and Cisco Express Forwarding as a backup or alternate path, and CEF programs it into line cards.

The BGP PIC feature provides the ability for CEF to quickly switch the traffic to the other egress ports if the current next hop or the link to this next hop goes down.

Figure 5: BGP PIC Edge and BGP FRR


BGP IPv4 NLRI
(IPv4 unicast and global address family)
BGP VPNv4 NLRI
(VPNv4 and VRF address family)
BGP FRR
Selects and stores best path and backup/alternate path per prefix.
BGP RIB
IP RIB
PIC Software
Reacts to next hop failure and switches to alternate path. Convergence not dependent on number of prefixes.
FIB
PIC Hardware
CEF/FIB H/W API
DFIB
DFIB
. . .
196159

BGP PIC over BDI with FRR

Starting with Cisco IOS XE Release 3.11S, BGP PIC is supported for bridge domain interfaces with FRR on the Cisco ASR 903 router. For information on configuring bridge domains, see Configuring Ethernet Virtual Connections on the Cisco ASR 903 Router. For information on configuring LFA FRR or remote LFA FRR, see Configuring Loop-Free Alternate Fast Reroute on the Cisco ASR 903 Router.

Detect a Failure

IGP detects a failure in the iBGP (remote) peer; it may take a few seconds to detect the failure. Convergence can occur in subseconds or seconds, depending on whether PIC is enabled on the line cards.

If the failure is among the directly connected neighbors (eBGP), and if you use BFD to detect when a neighbor has gone down. Depending on whether PIC is enabled on the line cards, the detection may happen within subseconds and the convergence can occur in subseconds or few seconds.

How BGP PIC Achieves Subsecond Convergence

The Cisco router supports subsecond convergence for up to 4000 prefixes (including global and VPN prefixes) during both BGP PIC core and edge failures.

MPLS VPN–BGP Local Convergence

The BGP PIC is an enhancement to the MPLS VPN–BGP Local Convergence feature. It provides a failover mechanism that recalculates the best path after a link failure. It then installs the new path in forwarding. To minimize traffic loss, the feature maintains the local label for 5 minutes to ensure that the traffic uses the backup or alternate path.

The BGP PIC improves the LoC time to under a second by calculating a backup or alternate path in advance. When a link failure occurs, the traffic is sent to the backup or alternate path.

When you configure BGP PIC, it overrides the functionality of the MPLS VPN--BGP Local Convergence feature. Do not remove the **protection local-prefixes** command from the configuration.

Overview of IPv6 VPN Provider Edge (6PE/VPE)

Note 6PE/VPE is supported on the RSP3 module starting with Cisco IOS XE Release 3.18SP.

IPv6 VPN Provider Edge (6PE/VPE) uses the existing MPLS IPv4 core infrastructure for IPv6 transport. 6PE/VPE enables IPv6 sites to communicate with each other over an MPLS IPv4 core network using MPLS label switched paths (LSPs). This feature relies on multiprotocol Border Gateway Protocol (BGP) extensions in the IPv4network configuration on the provider edge (PE) router to exchange IPv6 reachability information, in addition to an MPLS label) for each IPv6 address prefix. Edge routers are configured as dual-stack, running both IPv4 and IPv6, and use the IPv4 mapped IPv6 address for IPv6 prefix reachability exchange.

A hierarchy of labels are imposed on the 6PE ingress routers to keep the IPv6 traffic transparent to all the core routers. The top label provides connectivity inside the IPv4 MPLS core network and the label is distributed by Label Distribution Protocol (LDP). The bottom label, assigned to the IPv6 prefix of the destination, is distributed by multiprotocol BGP and used at each 6PE egress router for IPv6 forwarding.

All 6PE and core routers within the MPLS domain share a common IPv4 Interior Gateway Protocol (IGP) such as Open Shortest Path First (OSPF) or Integrated Intermediate System-to-Intermediate System (IS-IS).

IPv6 VPN uses the coexistence between IPv6 and IPv4 by leveraging an existent MPLS IPv4 core network. This approach is called 6VPE. The routing component of the VPN operation is divided into core routing and edge routing.

- Core routing, which involves PE routers and P routers, typically is performed by an IPv4 Interior Gateway Protocol (IGP) such as Open Shortest Path First (OSPF) or Intermediate System-to-Intermediate System (IS-IS).
- Edge routing takes place in two directions—routing between PE pairs and routing between a PE and a CE.

  Routing between PE pairs is achieved using multiprotocol internal BGP (iBGP) using the IPv6 VPN address family. This method distributes routes learned from CEs through PE-CE routing, using appropriate route export policies at the ingress PE router and appropriate route import policies at the egress PE router.

Routing between the CE and its PE is achieved using a routing protocol that is VPN routing and forwarding (VRF) aware.

Static routes and external BGP (eBGP) are VRF instance aware.

BGP PIC Scenario

You can configure the BGP PIC functionality to achieve fast convergence.

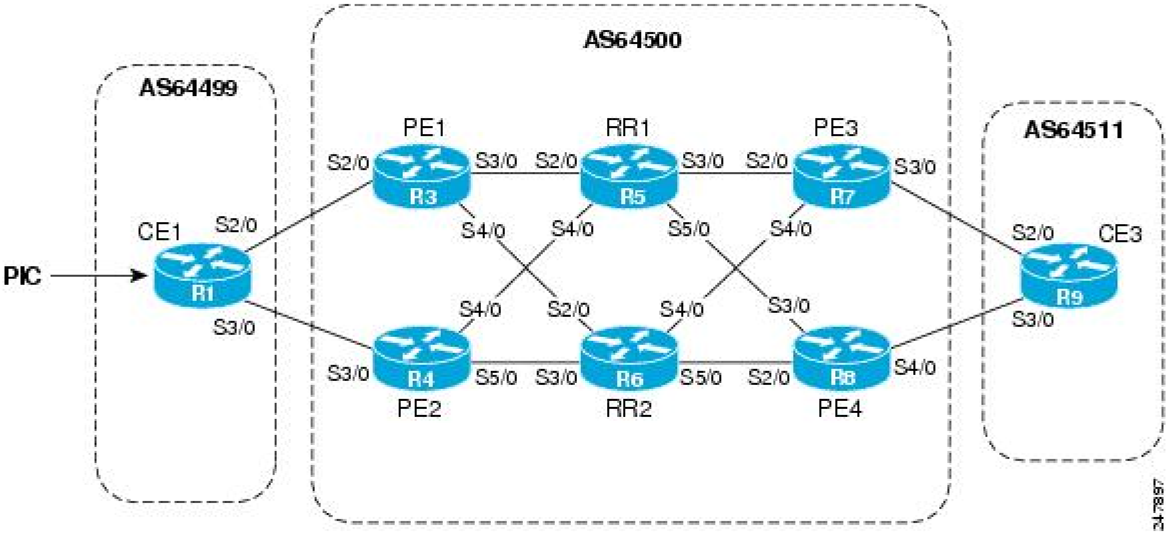
IP PE-CE Link and Node Protection

The network includes the following components:

- eBGP sessions exist between the PE and CE routers.
- Traffic from CE1 uses PE1 to reach network 192.168.9.0/24 through router CE3.
- CE1 has two paths:
  - PE1 as the primary path.
  - PE2 as the backup or alternate path.

CE1 is configured with the BGP PIC feature. BGP computes PE1 as the best path and PE2 as the backup or alternate path. It installs both routes into the RIB and Cisco Express Forwarding plane. When the CE1-PE1 link goes down, Cisco Express Forwarding detects the link failure and points the forwarding object to the backup or alternate path. Traffic is quickly rerouted due to local fast convergence in Cisco Express Forwarding.

Figure 6: Using BGP PIC to Protect the PE-CE Link



IP PE-CE Link and Node Protection on the CE Side (Dual CEs and Dual PE Primary and Backup Nodes)

The figure below shows a network that uses the BGP PIC feature on CE1. The network includes the following components:

- eBGP sessions exist between the PE and CE routers.

- Traffic from CE1 uses PE1 to reach network 192.168.9.0/24 through router CE3.
- CE1 has two paths:
    - PE1 as the primary path.
    - PE2 as the backup/alternate path.
- An iBGP session exists between the CE1 and CE2 routers.

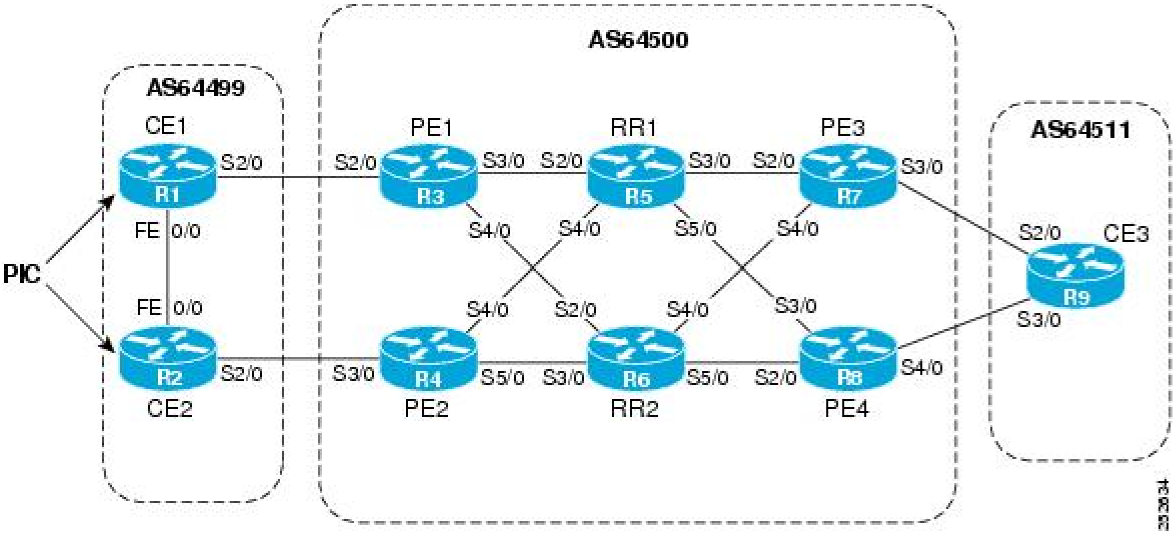
In this example, CE1 and CE2 are configured with the BGP PIC feature. BGP computes PE1 as the best path and PE2 as the backup/alternate path and installs both the routes into the RIB and Cisco Express Forwarding plane.

There should not be any policies set on CE1 and CE2 for the eBGP peers PE1 and PE2. Both CE routers must point to the eBGP route as next hop. On CE1, the next hop to reach CE3 is through PE1, so PE1 is the best path to reach CE3. On CE2, the best path to reach CE3 is PE2. CE2 advertises itself as the next hop to CE1, and CE1 does the same to CE2. As a result, CE1 has two paths for the specific prefix and it usually selects the directly connected eBGP path over the iBGP path according to the best path selection rules. Similarly, CE2 has two paths--an eBGP path through PE2 and an iBGP path through CE1-PE1.

When the CE1-PE1 link goes down, Cisco Express Forwarding detects the link failure and points the forwarding object to the backup/alternate node CE2. Traffic is quickly rerouted due to local fast convergence in Cisco Express Forwarding.

If the CE1-PE1 link or PE1 goes down and BGP PIC is enabled on CE1, BGP recomputes the best path, removing the next hop PE1 from RIB and reinstalling CE2 as the next hop into the RIB and Cisco Express Forwarding. CE1 automatically gets a backup/alternate repair path into Cisco Express Forwarding and the traffic loss during forwarding is now in subseconds, thereby achieving fast convergence.

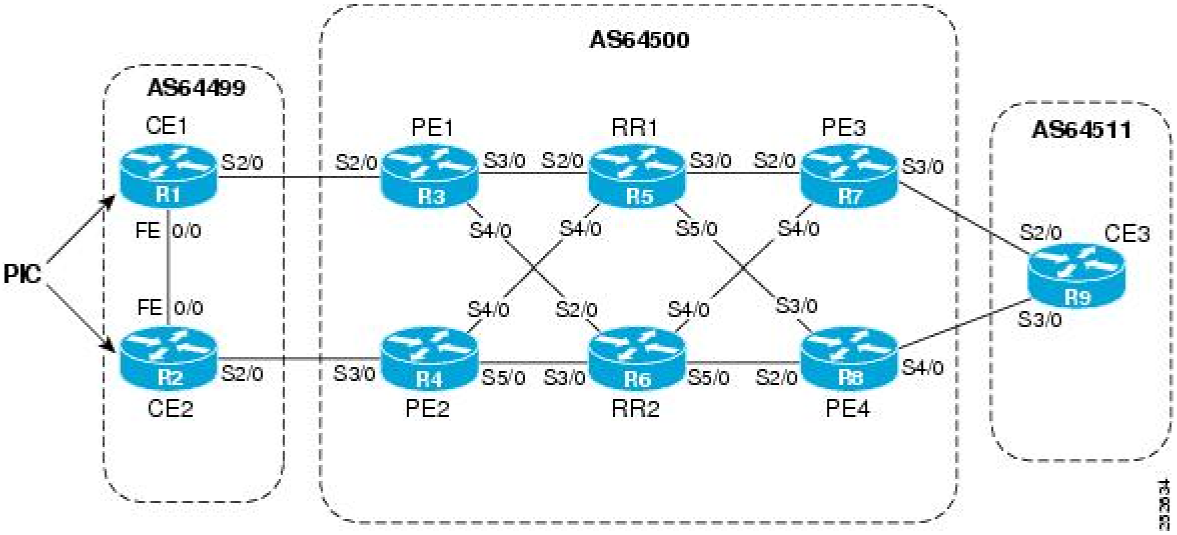
Figure 7: Using BGP PIC in a Dual CE, Dual PE Network



IP MPLS PE-CE Link Protection for the Primary or Backup-Alternate Path

The figure below shows a network that uses the BGP PIC feature on CE1 and CE2. The network includes the following components:

Figure 8: Using BGP PIC in a Dual CE, Dual PE Network



- eBGP sessions exist between the PE and CE routers.
- The PE routers are VPNv4 iBGP peers with reflect routers in the MPLS network.
- Traffic from CE1 uses PE1 to reach the network 192.168.9.0/24 through router CE3.
- CE3 is dual-homed with PE3 and PE4.
- PE1 has two paths to reach CE3 from the reflect routers:
    - PE3 is the primary path with the next hop as a PE3 address.
    - PE4 is the backup/alternate path with the next hop as a PE4 address.

In this example, all the PE routers can be configured with the BGP PIC feature under IPv4 or VPNv4 address families.

For BGP PIC to work in BGP for PE-CE link protection, set the policies on PE3 and PE4 for prefixes received from CE3 so that one of the PE routers acts as the primary and the other as the backup/alternate. Usually, this is done using local preference and giving better local preference to PE3. In the MPLS cloud, traffic internally flows through PE3 to reach CE3. Thus, PE1 has PE3 as the best path and PE4 as the second path.

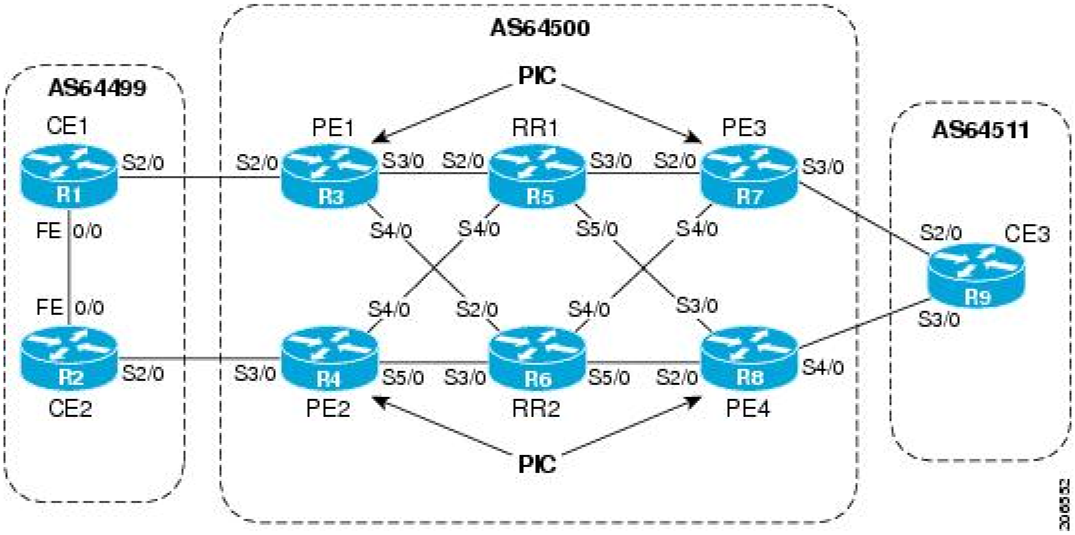
When the PE3-CE3 link goes down, Cisco Express Forwarding detects the link failure, and PE3 recomputes the best path, selects PE4 as the best path, and sends a withdraw message for the PE3 prefix to the reflect routers. Some of the traffic goes through PE3-PE4 until BGP installs PE4 as the best path route into the RIB and Cisco Express Forwarding. PE1 receives the withdraw, recomputes the best path, selects PE4 as the best path, and installs the routes into the RIB and Cisco Express Forwarding plane.

Thus, with BGP PIC enabled on PE3 and PE4, Cisco Express Forwarding detects the link failure and does in-place modification of the forwarding object to the backup/alternate node PE4 that already exists in Cisco Express Forwarding. PE4 knows that the backup/alternate path is locally generated and routes the traffic to the egress port connected to CE3. This way, traffic loss is minimized and fast convergence is achieved.

IP MPLS PE-CE Node Protection for Primary or Backup-Alternate Path

The figure below shows a network that uses the BGP PIC feature on all the PE routers in an MPLS network.

Figure 9: Enabling BGP PIC on All PEs Routers in the MPLS Network



The network includes the following components:

- eBGP sessions exist between the PE and CE routers.
- The PE routers are VPNv4 iBGP peers with reflect routers in the MPLS network.
- Traffic from CE1 uses PE1 to reach the network 192.168.9.0/24 through router CE3.
- CE3 is dual-homed with PE3 and PE4.
- PE1 has two paths to reach CE3 from the reflect routers:
  - PE3 is the primary path with the next hop as a PE3 address.
  - PE4 is the backup/alternate path with the next hop as a PE4 address.

In this example, all the PE routers are configured with the BGP PIC feature under IPv4 and VPNv4 address families.

For BGP PIC to work in BGP for the PE-CE node protection, set the policies on PE3 and PE4 for the prefixes received from CE3 such that one of the PE routers acts as primary and the other as backup/alternate. Usually, this is done using local preference and giving better local preference to PE3. In the MPLS cloud, traffic internally flows through PE3 to reach CE3. So, PE1 has PE3 as the best path and PE4 as the second path.

When PE3 goes down, PE1 knows about the removal of the host prefix by IGPs in subseconds, recomputes the best path, selects PE4 as the best path, and installs the routes into the RIB and Cisco Express Forwarding plane. Normal BGP convergence will happen while BGP PIC is redirecting the traffic through PE4, and packets are not lost.

Thus, with BGP PIC enabled on PE3, Cisco Express Forwarding detects the node failure on PE3 and points the forwarding object to the backup/alternate node PE4. PE4 knows that the backup/alternate path is locally generated and routes the traffic to the egress port using the backup/alternate path. This way, traffic loss is minimized.

No Local Policies Set on the PE Routers

PE1 and PE2 point to the eBGP CE paths as the next hop with no local policy. Each of the PE routers receives the other's path, and BGP calculates the backup/alternate path and installs it into Cisco Express Forwarding, along with its own eBGP path towards CE as the best path. The limitation of the MPLS PE-CE link and node protection solutions is that you cannot change BGP policies. They should work without the need for a best-external path.

Local Policies Set on the PE Routers

Whenever there is a local policy on the PE routers to select one of the PE routers as the primary path to reach the egress CE, the **bgp advertise-best-external** command is needed on the backup/alternate node PE3 to propagate the external CE routes with a backup/alternate label into the route reflectors and the far-end PE routers.

Enable BGP PIC

Because many service provider networks contain many VRFs, the BGP PIC allows you to configure, at a time, the BGP PIC feature for all VRFs.

- VPNv4 address family configuration mode protects all VRFs.
- VRF-IPv4 address family configuration mode protects only IPv4 VRFs.
- Router configuration mode protects prefixes in the global routing table.

Cisco Express Forwarding Recursion

Recursion is the ability to find the next longest matching path when the primary path goes down.

If BGP PIC is not installed, and if the next hop to a prefix fails, Cisco Express Forwarding finds the next path to reach the prefix by recursing through the FIB to find the next longest matching path to the prefix. This recurstion mechanism is useful when the next hop is multiple hops away and there is more than one way of reaching the next hop.

However, with the BGP PIC feature, you may want to disable Cisco Express Forwarding recursion for the following reasons:

- Recursion slows down convergence when Cisco Express Forwarding searches all the FIB entries.
- BGP PIC Edge already precomputes an alternate path. It therefore eliminates the need for Cisco Express Forwarding recursion.

When the BGP PIC functionality is enabled, Cisco Express Forwarding recursion is disabled by default for two conditions:

- For next hops learned with a /32 network mask (host routes)
- For next hops that are directly connected.

For all other cases, Cisco Express Forwarding recursion is enabled.

You can issue the **bgp recursion host** command to disable or enable Cisco Express Forwarding recursion for BGP host routes. This provision is part of the BGP PIC functionality.

Note When the BGP PIC feature is enabled, by default, **bgp recursion host** is configured for VPNv4 and VPNv6 address families and disabled for IPv4 and IPv6 address families.

To disable or enable Cisco Express Forwarding recursion for BGP directly connected next hops, run the **disable-connected-check** command.

How to Configure BGP PIC

Configuring BGP PIC

Because many service provider networks contain many VRFs, the BGP PIC feature allows you to configure the BGP PIC feature for all VRFs at once.

- VPNv4 address family configuration mode protects all the VRFs.
- VRF-IPv4 address family configuration mode protects only IPv4 VRFs.
- Router configuration mode protects prefixes in the global routing table.

For a full configuration example that includes configuring multiprotocol VRFs and shows output to verify that the feature is enabled, see the Example: Configuring BGP PIC.

Before you begin

- If you are implementing the BGP PIC feature in an MPLS VPN, ensure that the network is working properly before configuring the BGP PIC feature. See the *MPLS: Layer 3 VPNs Configuration Guide* for more information.
- If you are implementing the BGP PIC feature in an MPLS VPN, configure multiprotocol VRFs, which allow you to share route-target policies (import and export) between IPv4 and IPv6 or to configure separate route-target policies for IPv4 and IPv6 VPNs. For information about configuring multiprotocol VRFs, see MPLS VPN--VRF CLI for IPv4 and IPv6 VPNs .
- Ensure that the CE router is connected to the network by at least two paths.

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.

Step 4	Do one of the following:
	• **address-family ipv4** [**unicast** | **vrf** *vrf-name*]
	• or
	• **address-family vpnv4** [**unicast**]
	Example:
	`Device(config-router)# address-family ipv4 unicast`
	Example:
	`Device(config-router)# address-family vpnv4`
		Specifies the IPv4 or VPNv4 address family and enters address family configuration mode.
		• The **unicast** keyword specifies the IPv4 or VPNv4 unicast address family.
		• The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.

Step 5	**bgp additional-paths install** **Example:** `Device(config-router-af)# bgp additional-paths install`	Calculates a backup/alternate path and installs it into the RIB and Cisco Express Forwarding.
Step 6	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types.
Step 7	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local router.
Step 8	**bgp recursion host** **Example:**	(Optional) Enables the recursive-via-host flag for IPv4, VPNv4, and VRF address families.

	Command or Action	Purpose
	`Device(config-router-af)# bgp recursion host`	• When the BGP PIC feature is enabled, Cisco Express Forwarding recursion is disabled. Under most circumstances, you do not want to enable recursion when BGP PIC is enabled.
Step 9	**neighbor** *ip-address* **fall-over** [**bfd** |**route-map** *map-name*] **Example:** `Device(config-router-af)# neighbor 192.168.1.1 fall-over bfd`	Enables BFD protocol support to detect when a neighbor has gone away, which can occur within a subsecond.

Step 10	**end**
	Example:
	`Device(config-router-af)# end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Disabling BGP PIC Core

BGP PIC core feature is enabled by default. Use the following configuration to disable the BGP PIC core feature.

Note Use the **cef table output-chain build favor convergence-speed** command in global configuration mode to re-enable the BGP PIC core feature.

Procedure

	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	**cef table output-chain build favor memory-utilization** **Example:**	Configures memory characteristics for Cisco Express Forwarding table output chain building for the forwarding of packets through the network.

	Command or Action	Purpose
	`Device(config)# cef table output-chain build favor memory-utilization`	

Step 4	**end**
	Example:
	`Device(config)# end`
		Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP PIC

Example: Configuring BGP PIC

The following example shows how to configure the BGP PIC feature in VPNv4 address family configuration mode, which enables the feature on all VRFs. In the following example, there are two VRFs defined: blue and green. All the VRFs, including those in VRFs blue and green, are protected by backup/alternate paths.

```
vrf definition test1
 rd 400:1
 route-target export 100:1
 route-target export 200:1
 route-target export 300:1
 route-target export 400:1
 route-target import 100:1
 route-target import 200:1
 route-target import 300:1
 route-target import 400:1
 address-family ipv4
 exit-address-family
 exit
!
interface GigabitEthernet 0/0/0
 vrf forwarding test1
 ip address 10.0.0.1 255.0.0.0
 exit
router bgp 3
 no synchronization
 bgp log-neighbor-changes
 redistribute static
 redistribute connected
 neighbor 10.6.6.6 remote-as 3
 neighbor 10.6.6.6 update-source Loopback0
 neighbor 10.7.7.7 remote-as 3
 neighbor 10.7.7.7 update-source Loopback0
 no auto-summary
 !
 address-family vpnv4
  bgp additional-paths install
  neighbor 10.6.6.6 activate
  neighbor 10.6.6.6 send-community both
  neighbor 10.7.7.7 activate
  neighbor 10.7.7.7 send-community both
 exit-address-family
 !
 address-family ipv4 vrf blue
  import path selection all
  import path limit 10
  no synchronization
  neighbor 10.11.11.11 remote-as 1
  neighbor 10.11.11.11 activate
 exit-address-family
 !
 address-family ipv4 vrf green
  import path selection all
  import path limit 10
  no synchronization
  neighbor 10.13.13.13 remote-as 1
  neighbor 10.13.13.13 activate
 exit-address-family
```

The following **show vrf detail** command output shows that the BGP PIC feature is enabled:

```
Router# show vrf detail
VRF test1 (VRF Id = 1); default RD 400:1; default VPNID <not set>
  Interfaces:
    Se4/0
Address family ipv4 (Table ID = 1 (0x1)):
  Export VPN route-target communities
    RT:100:1                 RT:200:1                 RT:300:1
    RT:400:1
  Import VPN route-target communities
    RT:100:1                 RT:200:1                 RT:300:1
    RT:400:1
  No import route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
  Prefix protection with additional path enabled
Address family ipv6 not active.
```

Example: Configuring IPv6 BGP PIC Edge

On Primary PE

```
router bgp 100
address-family ipv6 vrf V1
 bgp  additional-paths install
 neighbor 2.2.2.2 activate
  neighbor 2.2.2.2 send-community both
  neighbor 2.2.2.2 next-hop-self
  neighbor 2.2.2.2 send-label
  exit-address-family
```

On Backup PE

```
router bgp 100
address-family ipv6 vrf V1
bgp advertise-best-external
neighbor 10.0.0.1 activate
neighbor 10.0.0.1 send-community both
neighbor 10.0.0.1 next-hop-self
neighbor 10.0.0.1 send-label
```

Example: Disabling BGP PIC Core

The following example shows how to disable the BGP PIC core in global configuration mode.

```
Device> enable
Device# configure terminal
Device(config)# cef table output-chain build favor memory-utilization
Device(config)# end
```

Verification Examples for BGP PIC

Example: Displaying Backup Alternate Paths for BGP PIC

The command output in the following example shows that the VRFs in VRF blue have backup/alternate paths:

```
Device# show ip bgp vpnv4 vrf blue 10.0.0.1

BGP routing table entry for 10:12:10.0.0.1/24, version 88
Paths: (4 available, best #1, table blue)
  Additional-path
  Advertised to update-groups:
     6
  1, imported path from 12:23:10.0.0.1/24
    10.3.3.3 (metric 21) from 10.6.6.6 (10.6.6.6)
      Origin incomplete, metric 0, localpref 200, valid, internal, best
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
  1, imported path from 12:23:10.0.0.1/24
    10.13.13.13 (via green) from 10.13.13.13 (10.0.0.2)
      Origin incomplete, metric 0, localpref 100, valid, external
      Extended Community: RT:12:23 , recursive-via-connected
  1, imported path from 12:23:10.0.0.1/24
    10.3.3.3 (metric 21) from 10.7.7.7 (10.7.7.7)
      Origin incomplete, metric 0, localpref 200, valid, internal
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
  1
    10.11.11.11 from 10.11.11.11 (1.0.0.1)
      Origin incomplete, metric 0, localpref 100, valid, external, backup/repair
      Extended Community: RT:11:12 , recursive-via-connected
```

The command output in the following example shows that the VRFs in VRF green have backup/alternate paths:

```
Device# show ip bgp vpnv4 vrf green 10.0.0.1

BGP routing table entry for 12:23:10.0.0.1/24, version 87
Paths: (4 available, best #4, table green)
  Additional-path
  Advertised to update-groups:
     5
  1, imported path from 11:12:10.0.0.1/24
    10.11.11.11 (via blue) from 10.11.11.11 (1.0.0.1)
```

```
      Origin incomplete, metric 0, localpref 100, valid, external
      Extended Community: RT:11:12 , recursive-via-connected
  1
    10.3.3.3 (metric 21) from 10.7.7.7 (10.7.7.7)
      Origin incomplete, metric 0, localpref 200, valid, internal
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
  1
    10.13.13.13 from 10.13.13.13 (10.0.0.2)
      Origin incomplete, metric 0, localpref 100, valid, external, backup/repair
      Extended Community: RT:12:23 , recursive-via-connected
  1
    10.3.3.3 (metric 21) from 10.6.6.6 (10.6.6.6)
      Origin incomplete, metric 0, localpref 200, valid, internal, best
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
```

The command output in the following example shows the BGP routing table entries for the backup and alternate paths:

```
Device# show ip bgp 10.0.0.1 255.255.0.0

BGP routing table entry for 10.0.0.1/16, version 123
Paths: (4 available, best #3, table default)
  Additional-path
  Advertised to update-groups:
     2          3
  Local
    10.0.101.4 from 10.0.101.4 (10.3.3.3)
      Origin IGP, localpref 100, weight 500, valid, internal
  Local
    10.0.101.3 from 10.0.101.3 (10.4.4.4)
      Origin IGP, localpref 100, weight 200, valid, internal
  Local
    10.0.101.2 from 10.0.101.2 (10.1.1.1)
      Origin IGP, localpref 100, weight 900, valid, internal, best
  Local
    10.0.101.1 from 10.0.101.1 (10.5.5.5)
      Origin IGP, localpref 100, weight 700, valid, internal, backup/repair
```

The command output in the following example shows the routing information base entries for the backup and alternate paths:

```
Device# show ip route repair-paths 10.0.0.1 255.255.0.0

Routing entry for 10.0.0.1/16
  Known via "bgp 10", distance 200, metric 0, type internal
  Last update from 10.0.101.2 00:00:56 ago
  Routing Descriptor Blocks:
  * 10.0.101.2, from 10.0.101.2, 00:00:56 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
      MPLS label: none
    [RPR]10.0.101.1, from 10.0.101.1, 00:00:56 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
      MPLS label: none
```

The command output in the following example shows the Cisco Express Forwarding/forwarding information base entries for the backup and alternate paths:

```
Device# show ip cef 10.0.0.1 255.255.0.0 detail

10.0.0.1/16, epoch 0, flags rib only nolabel, rib defined all labels
  recursive via 10.0.101.2
    attached to GigabitEthernet0/0/2
  recursive via 10.0.101.1, repair
    attached to GigabitEthernet0/0/2
```

Example: Verifying BGP PIC Edge

show ip bgp all summary

```
Router# show ip bgp all summary

For address family: IPv4 Unicast
BGP router identifier 65.1.160.1, local AS number 100
BGP table version is 1, main routing table version 1
 Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
3.3.3.3         4          100     154      50        1    0    0 00:33:33        0
4.4.4.4         4          100   16158   10579        1    0    0 6d14h           0

For address family: IPv6 Unicast
BGP router identifier 65.1.160.1, local AS number 100
BGP table version is 363156, main routing table version 363156

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
3.3.3.3         4          100     154      50   363156    0    0 00:33:33        0
4.4.4.4         4          100   16158   10579   363156    0    0 6d14h           0
For address family: VPNv4 Unicast
BGP router identifier 65.1.160.1, local AS number 100
BGP table version is 120, main routing table version 120
1 network entries using 156 bytes of memory
1 path entries using 80 bytes of memory
1/1 BGP path/bestpath attribute entries using 168 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
1 BGP extended community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 452 total bytes of memory
BGP activity 19514/15612 prefixes, 121023/113221 paths, scan interval 60 secs
 Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
3.3.3.3         4          100     154      50      120    0    0 00:33:34        1
4.4.4.4         4          100   16158   10579      120    0    0 6d14h           0
For address family: VPNv6 Unicast
BGP router identifier 65.1.160.1, local AS number 100
BGP table version is 291083, main routing table version 291083
3901 network entries using 702180 bytes of memory
7801 path entries using 842508 bytes of memory
2/2 BGP path/bestpath attribute entries using 336 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
1 BGP extended community entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 1545072 total bytes of memory
BGP activity 19514/15612 prefixes, 121023/113221 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down State/PfxRcd
3.3.3.3         4          100     154      50   291083    0    0 00:33:35     3900
4.4.4.4         4          100   16159   10579   291083    0    0 6d14h        3900
```

show ipv6 route vrf

```
Router# show ipv6 route vrf vpn1 800::0/64

Routing entry for 800::/64
  Known via "bgp 100", distance 200, metric 0, type internal
  Route count is 1/1, share count 0
  Routing paths:
    3.3.3.3%default indirectly connected [Host Res]
      MPLS label: 166
      Last updated 00:43:36 ago
    4.4.4.4%default indirectly connected [Repair] [Host Res]
      MPLS label: 7943
      Last updated 01:08:22 ago
```

show ipv6 cef

```
Router# show ipv6 cef 900::0 in

900::/64, epoch 2, flags [rlbls], RIB[B], refcnt 5, per-destination sharing
  sources: RIB
  feature space:
    IPRM: 0x00018000
    Broker: linked, distributed at 4th priority
    LFD: 900::/64 0 local labels
        contains path extension list
  ifnums: (none)
  path list 3C5993C8, 3801 locks, per-destination, flags 0x34D [shble, hvsh, rif, hwcn,
bldmp, bgp]
    path 3C594928, share 1/1, type recursive, for IPv6, flags [must-be-lbld]
      MPLS short path extensions: MOI flags = 0x0 label 2316
      recursive via 3.3.3.3[IPv4:Default] label 2316, fib 3D36FC7C, 1 terminal fib,
v4:Default:3.3.3.3/32
      path list 3C599968, 127 locks, per-destination, flags 0x4D [shble, hvsh, rif, hwcn]
          path 3C594ED8, share 1/1, type attached nexthop, for IPv4
            MPLS short path extensions: MOI flags = 0x0 label 84
            nexthop 21.1.1.2 GigabitEthernet0/1/3  label 84, IP adj out of
GigabitEthernet0/1/3  , addr 21.1.1.2 3CCB3320
 path 3C594AC8, share 1/1, type recursive, for IPv6, flags [must-be-lbld, rpr]
      MPLS short path extensions: MOI flags = 0x0 label 57
      recursive via 4.4.4.4[IPv4:Default] label 57, repair, fib 3D4BCD74, 1 terminal fib,
v4:Default:4.4.4.4/32
      path list 3C599668, 129 locks, per-destination, flags 0x4D [shble, hvsh, rif, hwcn]
          path 3C594B98, share 1/1, type attached nexthop, for IPv4
            MPLS short path extensions: MOI flags = 0x0 label 19
            nexthop 21.1.6.1 GigabitEthernet0/0/0  label 19, IP adj out of
GigabitEthernet0/0/0, addr 21.1.6.1 3D4D8BE0
  output chain:
loadinfo 3DCFE310, per-session, 2 choices, flags 0005, 4 locks
      flags [Per-session, for-rx-IPv6]
      translation map 3CF06908 owned by path list 3C5993C8, 1902 locks
        2 choices, 16 buckets, flags 0x1
        Path index      [  0  1  0  1  0  1  0  1  0  1  0  1  0  1  0  1 ]
        Repair path     [  -  X  -  X  -  X  -  X  -  X  -  X  -  X  -  X ]

        Path available  [  X  X  X  X  X  X  X  X  X  X  X  X  X  X  X  X ]
        Translation map [  0  0  2  2  4  4  6  6  8  8 10 10 12 12 14 14 ]
      16 hash buckets
        < 0 > label 2316
              loadinfo 3D572F10, per-session, 1 choice, flags 0111, 3904 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
```

```
                  < 0 > label 84
                        TAG adj out of GigabitEthernet0/1/3 , addr 21.1.1.2 3CCB3180
                Subblocks:
                  None
        < 1 > label 2316
              loadinfo 3D572F10, per-session, 1 choice, flags 0111, 3904 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label 84
                        TAG adj out of GigabitEthernet0/1/3 , addr 21.1.1.2 3CCB3180
                Subblocks:
                  None
```

show vrf detail

```
Router# show vrf detail

VRF Mgmt-intf (VRF Id = 1); default RD <not set>; default VPNID <not set>
  New CLI format, supports multiple address-families
  Flags: 0x1808
  Interfaces:
    Gi0
Address family ipv4 unicast (Table ID = 0x1):
  Flags: 0x0
  No Export VPN route-target communities
  No Import VPN route-target communities
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
Address family ipv6 unicast (Table ID = 0x1E000001):
  Flags: 0x0
  No Export VPN route-target communities
  No Import VPN route-target communities
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
VRF label allocation mode: per-prefix
Address family ipv4 multicast not active

VRF vpn1 (VRF Id = 2); default RD 100:1; default VPNID <not set>
  New CLI format, supports multiple address-families
  Flags: 0x180C
  Interfaces:
    BD13
Address family ipv4 unicast (Table ID = 0x2):
  Flags: 0x8000
  Export VPN route-target communities
    RT:100:1                 RT:100:10
  Import VPN route-target communities
    RT:100:1                 RT:100:10
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
  Prefix protection with additional path enabled
Address family ipv6 unicast (Table ID = 0x1E000002):
  Flags: 0x8000
  Export VPN route-target communities
    RT:100:1                 RT:100:10
  Import VPN route-target communities
    RT:100:1                      RT:100:10
```

show platform hardware pp active feature cef database

```
Router# show platform hardware pp active feature cef database ipv6 800::0/64 0x1E000002

=== CEF Prefix ===
800::/64 -- next hop: UEA Load Balance (PI:0x10844c30, PD:0x1543d948)
                Route Flags: (0)
                Handles (PI:0x105f4d30) (PD:0x16042c80)
   HW Info:
    TCAM handle: 0x00000270    TCAM index: 0x000067af
    FID index  : 0x00008abb    EAID      : 0x00005608
    MET        : 0x0002a8aa    FID Count : 0x00000000
 === Load Balance OCE ===
   PI:0x10844c30, PD:0x1543d948
   FID Count: 0x00000001
      Load Balance HW Info:
     Hardware Index: 0
       FID Index: 0x00008abb    RW Index  : 0x00000000
       MET      : 0x0002a8aa    EAID      : 0x00005608
       EL3 Index : 0x00001529   EL2 Index : 0x00000000
       Hardware Rewrite Str: 00:00:00:00:00:00:00:00:00:00:00:00
 === Label OCE ===
  Label flags: 4
  Num Labels: 1

Out Labels: 166
  Out Backup Labels: 1048577
  Next OCE Type: Loadbalance OCE; Next OCE handle: 0x121452f8
 === Load Balance OCE ===
   PI:0x1067bf38, PD:0x121452f8
   FID Count: 0x00000001
      Load Balance HW Info:
     Hardware Index: 0
       FID Index: 0x0000609c    RW Index  : 0x00000000
       MET      : 0x0002a8a8    EAID      : 0x0000ee72
       EL3 Index : 0x00001527   EL2 Index : 0x00000000
       Hardware Rewrite Str: 00:00:00:00:00:00:00:00:00:00:00:00

=== Label OCE ===
**This is selected Label OCE**
  Label flags: 4
  Num Labels: 1
  Num Bk Labels: 1
  Out Labels: 18
  Out Backup Labels: 1048577
  Next OCE Type: Adjacency; Next OCE handle: 0x119f3800

=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 21.1.1.2
  Interface: GigabitEthernet0/1/3     Protocol: TAG
  mtu:1500, flags:0x0, fixups:0x0, encap_len:14
  Handles (adj_id:0x000000f9) (PI:0x105f18d8) (PD:0x119f3800)
  Rewrite Str: d0:c2:82:cc:a5:cc:d0:c2:82:16:c4:8c:88:47

  HW Info:
    FID index: 0x0000605a    EL3 index: 0x0000100c   EL2 index: 0x00000000
    El2RW    : 0x00000108    MET index: 0x00032029    EAID     : 0x00001012
    HW ADJ FLAGS: 0x40
    Hardware MAC Rewrite Str: 00:00:00:00:00:00:00:00:00:00:00:00
```

```
=== Label OCE ===
  Label flags: 4
  Num Labels: 1
  Num Bk Labels: 1
  Out Labels: 7943
  Out Backup Labels: 1048577
  Next OCE Type: Loadbalance OCE; Next OCE handle: 0x12124f18

=== Load Balance OCE ===
   PI:0x10833fe8, PD:0x12124f18
   FID Count: 0x00000001
      Load Balance HW Info:
     Hardware Index: 0
       FID Index: 0x0000615c    RW Index  : 0x00000000
       MET      : 0x0002a894    EAID      : 0x0000aac7
       EL3 Index : 0x0000152b   EL2 Index : 0x00000000
       Hardware Rewrite Str: 00:00:00:00:00:00:00:00:00:00:00:00

=== Label OCE ===
**This is selected Label OCE**
  Label flags: 4
  Num Labels: 1
  Num Bk Labels: 1
  Out Labels: 19
  Out Backup Labels: 1048577
  Next OCE Type: Adjacency; Next OCE handle: 0x1211a648

=== Adjacency OCE ===
  Adj State: COMPLETE(0)   Address: 21.1.6.1
  Interface: GigabitEthernet0/0/0   Protocol: TAG
  mtu:1500, flags:0x0, fixups:0x0, encap_len:14
  Handles (adj_id:0x0000110b) (PI:0x106a94f0) (PD:0x1211a648)
  Rewrite Str: d0:c2:82:17:71:00:d0:c2:82:16:c4:80:88:47

  HW Info:
    FID index: 0x00006fdf    EL3 index: 0x00001001    EL2 index: 0x00000000
    El2RW    : 0x0000010c    MET index: 0x0003202d    EAID     : 0x00001009
    HW ADJ FLAGS: 0x40
    Hardware MAC Rewrite Str: 00:00:00:00:00:00:00:00:00:00:00:00
```

Convergence Enhancements for L3VPN Traffic over BGP PIC Edge

Table 2: Feature History

Feature Name	Release Information	Description
Convergence Enhancements for L3VPN Traffic over BGP PIC Edge	Cisco IOS XE Bengaluru 17.4.1	The BGP PIC feature is enhanced to achieve the sub-second convergence for greater than 2000 VPNv4/v6 prefixes using the hardware assisted PIC Edge functionality. Prior to this release, the BGP PIC Edge convergence could only be observed in seconds. To enable the feature, use the following new command: **platform l3vpn pic-enable**

The BGP PIC feature is enhanced to achieve the sub-second convergence for greater than 2000 VPNv4/v6 prefixes using the hardware assisted PIC Edge functionality. Prior to Cisco IOS XE Bengaluru Release 17.14.1, the BGP PIC Edge for VPNv4/v6 prefixes was software assisted. As a result, during a PIC Edge cutover, the average convergence for the VPNv4/v6 prefixes was only in seconds, based on the number of prefixes due to software delay.

The sub-second L3VPN PIC Edge convergence of the VPNv4/v6 prefixes is achieved by advertising through the address-family IPv4. You must configure **bgp mpls-local-label** command under **router bgp** configuration. Also, the value of **bgp nexthop trigger delay** command under VPNv4/v6 address family must be greater than the value of the trigger delay under address family IPv4.

Restrictions

- The implementation of the feature reduces the throughput of the L3VPN traffic to half.
- The IP Time-to-Live (TTL) of the packets is decremented twice.
- Only 1000 PIC Edge LBs are supported.
- You should not configure the **next-hop-self all** command under address-family ipv6 on RR to achieve sub-second convergence for 6PE prefixes.

Configuration Example: Convergence Enhancements for L3VPN Traffic over BGP PIC Edge

The following example shows how to configure convergence enhancements for L3VPN traffic over BGP PIC Edge:

```
enable
configure terminal
vrf definition test_vpn
rd 10:1
!
address-family ipv4
route-target export 10:1
route-target import 10:1
bgp next-hop Loopback1
exit-address-family
!
address-family ipv6
route-target export 10:1
route-target import 10:1
exit-address-family
platform l3vpn pic-enable
router bgp 100
 bgp log-neighbor-changes
 bgp ha-mode sso
 neighbor Acc peer-group
 neighbor Acc remote-as 100
 neighbor Acc update-source Loopback0
 neighbor 2.2.2.2 peer-group Acc
 neighbor 3.3.3.3 peer-group Acc
 !
 address-family ipv4
  bgp additional-paths install
  bgp nexthop trigger delay 1
  bgp mpls-local-label
  network 1.1.1.1 mask 255.255.255.255
  network 180.3.0.0
  neighbor Acc send-community both
  neighbor Acc send-label
  neighbor 2.2.2.2 activate
  neighbor 3.3.3.3 activate
 exit-address-family
 !
 address-family ipv4 multicast
 exit-address-family
 !
 address-family vpnv4
  neighbor Acc send-community extended
  neighbor 2.2.2.2 activate
neighbor 3.3.3.3 activate
 exit-address-family
 !
address-family ipv4 vrf test_vpn
neighbor 10.10.10.1 remote-as 200
neighbor 10.10.10.1 activate
redistribute connected
exit-address-family
!
address-family ipv4 vrf test_vpn1
exit-address-family
```

Verification of Convergence Enhancements for L3VPN Traffic over BGP PIC Edge Configuration

Use the following commands to verify the convergence enhancements for L3VPN traffic over BGP PIC Edge configuration:

```
Router#show ip cef vrf vrf1 150.1.1.1 detail
150.1.1.0/24, epoch 0, flags [rib defined all labels]
  recursive via 8.8.8.8 label 7140
    recursive via 4.4.4.4 label 4108-(local:417)
      nexthop 192.168.1.2 Port-channel1 label [19|45]-(local:18)
        repair: attached-nexthop 196.168.1.2 TenGigabitEthernet0/5/6
      nexthop 196.168.1.2 TenGigabitEthernet0/5/6, repair
    recursive via 7.7.7.7 label 4127-(local:417), repair
      nexthop 196.168.1.2 TenGigabitEthernet0/5/6 label [21|24]-(local:26)
        repair: attached-nexthop 192.168.1.2 Port-channel1
      nexthop 192.168.1.2 Port-channel1, repair
```

```
Router#show ip cef vrf vrf1 150.1.1.1 internal
150.1.1.0/24, epoch 0, flags [rlbls], RIB[B], refcnt 6, per-destination sharing
  sources: RIB
  feature space:
    IPRM: 0x00018000
    Broker: linked, distributed at 3rd priority
    LFD: 150.1.1.0/24 0 local labels
        contains path extension list
  ifnums: (none)
  path list 36A7E658, 8003 locks, per-destination, flags 0x24D [shble, hvsh, rif, hwcn,
bgp]
    path 35DE4D38, share 1/1, type recursive, for IPv4, flags [must-be-lbld]
      MPLS short path extensions: [rib] MOI flags = 0x1 label 7140
recursive via 8.8.8.8[IPv4:Default] label 7140, fib 33E56134, 1 terminal fib,
v4:Default:8.8.8.8/32
     path list 36A7DF98, 8015 locks, per-destination, flags 0x36D [shble, hvsh, rif, rcrsv,
 hwcn, bldmp, bgp]
          path 35DE81D0, share 1/1, type recursive, for IPv4
            MPLS short path extensions: [rib] MOI flags = 0x1 label 4108
            recursive via 4.4.4.4[IPv4:Default] label 4108-(local:417), fib 34B1AE14, 1
terminal fib, v4:Default:4.4.4.4/32
            path list 36A7E1D8, 3 locks, per-destination, flags 0x49 [shble, rif, hwcn]
                path 35DE4A08, share 1/1, type attached nexthop, for IPv4, flags [has-rpr]
                  MPLS short path extensions: [none] MOI flags = 0x21 label 19
                  nexthop 192.168.1.2 Port-channel1 label [19|45]-(local:18), IP adj out
of Port-channel1, addr 192.168.1.2 35F4E8E0
                   repair: attached-nexthop 196.168.1.2 TenGigabitEthernet0/5/6 (35DE7840)

                path 35DE7840, share 1/1, type attached nexthop, for IPv4, flags [rpr,
rpr-only]
                  nexthop 196.168.1.2 TenGigabitEthernet0/5/6, repair, IP adj out of
TenGigabitEthernet0/5/6, addr 196.168.1.2 351A6300
path 35DE9028, share 1/1, type recursive, for IPv4, flags [rpr]
            MPLS short path extensions: [rib] MOI flags = 0x1 label 4127
          recursive via 7.7.7.7[IPv4:Default] label 4127-(local:417), repair, fib 34B1B120,
 1 terminal fib, v4:Default:7.7.7.7/32
            path list 36A7E418, 3 locks, per-destination, flags 0x49 [shble, rif, hwcn]
                path 35DE50F0, share 1/1, type attached nexthop, for IPv4, flags [has-rpr]
                  MPLS short path extensions: [none] MOI flags = 0x21 label 21
                  nexthop 196.168.1.2 TenGigabitEthernet0/5/6 label [21|24]-(local:26), IP
 adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2 351A6300
                    repair: attached-nexthop 192.168.1.2 Port-channel1 (35DE8698)
                path 35DE8698, share 1/1, type attached nexthop, for IPv4, flags [rpr,
rpr-only]
                  nexthop 192.168.1.2 Port-channel1, repair, IP adj out of Port-channel1,
addr 192.168.1.2 35F4E8E0
  output chain:
    label 7140
loadinfo 34B543DC, per-session, 2 choices, flags 0511, 10205 locks
      flags [Per-session, for-mpls-not-at-eos, indirection, recursive]
      translation map 3628D4B0 owned by path list 36A7DF98, 4010 locks
        2 choices, 16 buckets, flags 0x1
        Path index      [  0  1  0  1  0  1  0  1  0  1  0  1  0  1  0  1 ]
        Repair path     [  -  X  -  X  -  X  -  X  -  X  -  X  -  X  -  X ]

        Path available  [  X  X  X  X  X  X  X  X  X  X  X  X  X  X  X  X ]
        Translation map [  0  0  2  2  4  4  6  6  8  8 10 10 12 12 14 14 ]
      16 hash buckets
        < 0 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
   flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                         <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                         <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
< 1 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
< 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                         <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                         <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
< 2 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
<primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>
                         <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
< 3 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                         <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                         <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
 Subblocks:
                  None
 < 4 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                         <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                         <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
```

```
< 5 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                        <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                        <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
        < 6 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                        <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                        <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
< 7 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
< 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
                        <primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>

                        <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
 < 8 > label 4108-(local:417)
              loadinfo 34B54554, per-session, 1 choice, flags 0111, 4011 locks
                flags [Per-session, for-mpls-not-at-eos, indirection]
                1 hash bucket
                  < 0 > label [19|45]-(local:18)
                        FRR Primary (0x373B1B00)
<primary: TAG adj out of Port-channel1, addr 192.168.1.2 35F4E6E0>
                        <repair:  TAG adj out of TenGigabitEthernet0/5/6, addr 196.168.1.2
 351A6100>
                Subblocks:
                  None
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
Basic MPLS VPNs	Configuring MPLS Layer 3 VPNs

Related Topic	Document Title
A failover feature that creates a new path after a link or node failure	MPLS VPN--BGP Local Convergence
Configuring multiprotocol VRFs	MPLS VPN--VRF CLI for IPv4 and IPv6 VPNs

MIBs

MIB	MIBs Link
None	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

RFCs

RFC	Title
RFC 1771	*A Border Gateway Protocol 4 (BGP-4)*
RFC 2547	*BGP/MPLS VPNs*

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

CHAPTER 4

BGP NSR Support for iBGP Peers

BGP NSR provides BGP nonstop routing (NSR) and nonstop forwarding (NSF) in the event of a switchover from an Active RP to the Standby RP. The BGP NSR Support for iBGP Peers feature provides NSR support for iBGP peers configured under the IPv4 unicast or IPv4 + label address family.

Restrictions on BGP NSR Support for iBGP Peers

- This feature applies to iBGP peers configured under IPv4 unicast or IPv4 + label address families.
- When you configure BGP with graceful restart and remove the BGP configuration using `no router bgp` command, the graceful restart timer starts. As a result, the stale entry is present in the BGP routing table and it is only removed after the BGP graceful restart timer is over.
- With BGP PIC edge enabled with additional paths on the system, the **ha-mode sso prefer** CLI command is not supported for the BGP neighbor and the opposite way.

Information About BGP NSR Support for iBGP Peers

Benefit of BGP NSR Support for iBGP Peers

Nonstop routing is beneficial for iBGP peers because it reduces the likelihood of dropped packets during switchover from the Active RP to the Standby RP. Switchover occurs when the Active RP fails for some reason, and the Standby RP takes control of Active RP operations.

How to Configure BGP NSR Support for iBGP Peers

Making an iBGP Peer NSR-Capable for the IPv4 Address Family

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 4000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv4** [**unicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. - The **unicast** keyword specifies the IPv4 unicast address family. - The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 4000`	Specifies the autonomous system of the neighbor.
Step 6	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 activate`	Activates the specified peer.

	Command or Action	Purpose
Step 7	**neighbor** *ip-address* **ha-mode sso** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 ha-mode sso`	Configures a BGP neighbor to support BGP NSR with stateful switchover (SSO).

Step 8	**end**
	Example:
	`Device(config-router-af)# end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Making an iBGP Peer NSR-Capable for the VPNv4 Address Family

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 4000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.1.1 remote-as 4000`	Specifies the autonomous system of the neighbor.
Step 5	**neighbor** *ip-address* **ha-mode sso** **Example:** `Device(config-router)# neighbor 192.168.1.1 ha-mode sso`	Configures a BGP neighbor to support BGP NSR with stateful switchover (SSO).
Step 6	**address-family vpnv4** [**unicast**] **Example:**	Specifies the VPNv4 address family and enters address family configuration mode.

	Command or Action	Purpose
	`Device(config-router)# address-family VPNv4 unicast`	
Step 7	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 activate`	Activates the specified peer.

Step 8	**end**
	Example:
	`Device(config-router-af)# end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Making an iBGP Peer NSR Capable at the Router Level

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 4000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 192.168.1.1 remote-as 4000`	Specifies the autonomous system of the neighbor.
Step 5	**neighbor** *ip-address* **activate** **Example:** `Device(config-router)# neighbor 192.168.1.1 activate`	Activates the specified neighbor.

	Command or Action	Purpose
Step 6	**neighbor** *ip-address* **ha-mode sso** **Example:** `Device(config-router)# neighbor 192.168.1.1 ha-mode sso`	Configures the specified peer to be NSR capable in all of the NSR-supported address families under which that peer has been activated.

Step 7	**end**
	Example:
	`Device(config-router)# end`
		Exits configuration mode and returns to privileged EXEC mode.

Step 8	**show ip bgp sso summary** **Example:** `Device# show ip bgp sso summary`	(Optional) Displays information about stateful switchover (sso) and whether a peer has NSR enabled or disabled.

Configuration Examples for BGP NSR Support for an iBGP Peer

Example: Configuring an iBGP Peer To Be NSR Capable

Configuring an iBGP Peer to Be NSR Capable at the Address Family Level

```
router bgp 4000
 address-family ipv4 unicast
 neighbor 192.168.1.1 remote-as 4000
 neighbor 192.168.1.1 activate
 neighbor 192.168.1.1 ha-mode sso
```

Configuring an iBGP Peer to Be NSR Capable at the Router Level

```
router bgp 4000
 neighbor 192.168.1.1 remote-as 4000
 neighbor 192.168.1.1 activate
 neighbor 192.168.1.1 ha-mode sso
```

Additional References

Related Documents

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

Related Topic	Document Title
BGP commands	Cisco IOS IP Routing: BGP Command Reference
BFD commands	Cisco IOS IP Routing: Protocol Independent Command Reference
Configuring BFD support for another routing protocol	IP Routing: BFD Configuration Guide

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

CHAPTER 5

BGP Graceful Shutdown

The BGP Graceful Shutdown feature reduces or eliminates the loss of traffic along a link being shut down for maintenance. Routers always have a valid route available during the convergence process. This feature is used primarily for maintenance on a link between a Provider Edge (PE), PE-PE, PE- Route Reflector (RR), PE-Customer Edge (CE) and CE.

Information About BGP Graceful Shutdown

Purpose and Benefits of BGP Graceful Shutdown

There are times when planned maintenance operations cause routing changes in BGP. After the shutdown of eBGP and iBGP peering sessions between autonomous system border routers (ASBRs), BGP devices are temporarily unreachable during BGP convergence. The goal of gracefully shutting down one or more BGP sessions is to minimize traffic loss during the planned shutdown and subsequent reestablishment of the sessions.

The BGP Graceful Shutdown feature reduces or eliminates the loss of inbound or outbound traffic flows that were initially forwarded along the peering link that is being shut down for maintenance. This feature is primarily for PE-CE, PE-RR and PE-PE links. Lowering the local preference for paths received over the session being shutdown renders the affected paths less preferred by the BGP decision process, but still allows the paths to be used during the convergence while alternative paths are propagated to the affected devices. Therefore, devices always have a valid route available during the convergence process.

The feature also allows vendors to provide a graceful shutdown mechanism that does not require any router reconfiguration at maintenance time. The benefits of the BGP Graceful Shutdown feature are fewer lost packets and less time spent reconfiguring devices.

GSHUT Community

The GSHUT community is a well-known community used in conjunction with the BGP Graceful Shutdown feature. The GSHUT community attribute is applied to a neighbor specified by the **neighbor shutdown graceful** command, thereby gracefully shutting down the link in an expected number of seconds. The GSHUT community is always sent by the GSHUT initiator.

The GSHUT community is specified in a community list, which is referenced by a route map and then used to make policy routing decisions.

The GSHUT community can also be used in the **show ip bgp community** command to limit output to GSHUT routes.

How to Configure BGP Graceful Shutdown

Shutting Down a BGP Link Gracefully

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 5000`	Configures a BGP routing process.
Step 4	**neighbor** {*ipv4-address* | *ipv6-address*} **remote-as** *number* **Example:** `Device(config-router)# neighbor 2001:db8:3::1 remote-as 5500`	Configures the autonomous system (AS) to which the neighbor belongs.

Step 5	**neighbor** {*ipv4-address* | *ipv6-address* | *peer-group-name*} **shutdown graceful** *seconds* {**community** *value* [**local-preference** *value*] | **local-preference** *value*}
	Example:
	`Device(config-router)# neighbor 2001:db8:3::1 shutdown graceful 600 community 1200 local-preference 300`
		Configures the device to gracefully shut down the link to the specified peer in the specified number of seconds; advertises the route with the GSHUT (Graceful Shutdown) community; and advertises the route with another community or specifies a local preference value for the route, or both.
		• Make sure to specify an adequate amount of time for iBGP peers to converge and to choose an alternate path as the best path.

	Command or Action	Purpose
		• If the **graceful** keyword is used in the **neighbor shutdown** command, at least one of the two attributes (a community or local preference) must be configured. You may configure both attributes. • If the **graceful** keyword is used in the **neighbor shutdown** command, the route is advertised with the GSHUT community by default. You may also set one other community for policy routing purposes. • In this particular example, the route to the neighbor is configured to shut down in 600 seconds, is advertised with the GSHUT community and community 1200, and is configured with a local preference of 300. • The device receiving the advertisement looks at the community value(s) of the route and optionally uses the community value to apply routing policy. Filtering routes based on a community is done with the **ip community-list** command and a route map. • During the graceful shutdown, the **neighbor shutdown** command is not nvgened. After the timer expires, SHUTDOWN is nvgened.

Step 6	**end**
	Example:
	`Device(config-router)# end`
		Returns to EXEC mode.

Step 7	**show ip bgp community gshut** **Example:** `Device# show ip bgp community gshut`	(Optional) Displays information about the routes that are advertised with the well-known GSHUT community.

Filtering BGP Routes Based on the GSHUT Community

Perform this task on a BGP peer to the device where you enabled the BGP Graceful Shutdown feature.

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 2000`	Configures a BGP routing process.
Step 4	**neighbor** {*ipv4-address* | *ipv6-address*} **remote-as** *number* **Example:** `Device(config-router)# neighbor 2001:db8:4::1 remote-as 1000`	Configures the autonomous system (AS) to which the neighbor belongs.
Step 5	**neighbor** {*ipv4-address* | *ipv6-address*} **activate** **Example:** `Device(config-router)# neighbor 2001:db8:4::1 activate`	Activates the neighbor.
Step 6	**neighbor** {*ipv4-address* | *ipv6-address*} **send-community** **Example:** `Device(config-router)# neighbor 2001:db8:4::1 send-community`	Enables BGP community exchange with the neighbor.
Step 7	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode.
Step 8	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map RM_GSHUT deny 10`	Configures a route map to permit or deny routes for policy routing.

	Command or Action	Purpose

Step 9	**match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}
	Example:
	`Device(config-route-map)# match community GSHUT`
		Configures that the routes that match ip community-list GSHUT will be policy routed.

Step 10	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode.
Step 11	**ip community-list** {*standard* | **standard** *list-name*} {**deny** | **permit**} **gshut** **Example:** `Device(config)# ip community-list standard GSHUT permit gshut`	Configures a community list and permits or denies routes that have the GSHUT community to the community list. • If you specify other communities in the same statement, there is a logical AND operation and all communities in the statement must match the communities for the route in order for the statement to be processed.
Step 12	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 2000`	Configures a BGP routing process.
Step 13	**neighbor** *address* **route-map** *map-name* **in** **Example:** `Device(config)# neighbor 2001:db8:4::1 route-map RM_GSHUT in`	Applies the route map to incoming routes from the specified neighbor. • In this example, the route map named RM_GSHUT denies routes from the specified neighbor that have the GSHUT community.

Configuration Examples for BGP Graceful Shutdown

Example: Shutting Down a BGP Link Gracefully

Graceful Shutdown While Setting a Local-Preference

This example gracefully shuts down the link to the specified neighbor in 600 seconds, adds the GSHUT community to the route, and sets a local preference of 500 for the route.

```
router bgp 1000
 neighbor 2001:db8:5::1 remote-as 2000
 neighbor 2001:db8:5::1 shutdown graceful 600 local-preference 500
 neighbor 2001:db8:5::1 send-community
 exit
```

Graceful Shutdown While Setting an Additional Community

This example gracefully shuts down the link to the specified neighbor in 600 seconds, and adds the GSHUT community and numbered community to the route.

```
router bgp 1000
 neighbor 2001:db8:5::1 remote-as 2000
 neighbor 2001:db8:5::1 shutdown graceful 600 community 1400
 neighbor 2001:db8:5::1 send-community
 exit
```

Graceful Shutdown while Setting an Additional Community and Local-Preference

This example gracefully shuts down the link to the specified neighbor in 600 seconds, adds the GSHUT community and the numbered community to the route, and sets a local preference of 500 to the route.

```
router bgp 1000
 neighbor 2001:db8:5::1 remote-as 2000
 neighbor 2001:db8:5::1 shutdown graceful 600 community 1400 local-preference 500
 neighbor 2001:db8:5::1 send-community
 exit
```

Example: Filtering BGP Routes Based on the GSHUT Community

In additional to being able to gracefully shut down a BGP route, another use of the GSHUT community is to configure a community list to filter routes with this community from getting into the BGP routing table.

This example illustrates how to use a community list to filter incoming BGP routes based on the GSHUT community. In this example, a route map named RM_GSHUT denies routes based on a standard community list named GSHUT. The community list contains routes with the GSHUT community. The route map is then applied to incoming routes from the neighbor at 2001:db8:4::1.

```
router bgp 2000
 neighbor 2001:db8:4::1 remote-as 1000
 neighbor 2001:db8:4::1 activate
 neighbor 2001:db8:4::1 send-community
 exit
route-map RM_GSHUT deny 10
 match community GSHUT
 exit
ip community-list standard GSHUT permit gshut
```

```
router bgp 2000
 neighbor 2001:db8:4::1 route-map RM_GSHUT in
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference

Standards and RFCs

Standard/RFC	Title
RFC 6198	*Requirements for the Graceful Shutdown of BGP Sessions*

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

CHAPTER 6

BFD—BGP Multihop Client Support, cBit (IPv4 and IPv6), and Strict Mode

The BFD—BGP Multihop Client Support feature enables Border Gateway Protocol (BGP) to use multihop Bidirectional Forwarding Detection (BFD) support, which improves BGP convergence as BFD detection and failure times are faster than the Interior Gateway Protocol (IGP) convergence times in most network topologies.

The BFD—BGP cBIT feature allows BGP to determine if BFD failure is dependent or independent of the Control Plane. This allows BGP greater flexibility in handling BFD down events.

Restrictions for BFD—BGP Multihop Client Support and cBit

- Since multihop BFD operates in software, the minimum timer recommended is 200 ms x 3.
- For BGP IPv4 and BGP IPv6 peering sessions only, multihop BFD support is available for BGP for address-family IPv4 and IPv6 unicast.
- For multihop BGP sessions using IPv6 Link Local addresses, BFD multihop support is not available.
- Currently BFD Hardware offload is not supported for multihop BFD sessions and so C-bit will not be set for multihop sessions.
- Multihop BFD for IPv6 Virtual Routing and Forwarding (VRF) is not supported.

Information About BFD - BGP Multihop Client Support and cBit (IPv4 and IPv6)

BFD—BGP Multihop Client Support, cBit (IPv4 and IPv6), and Strict Mode

BFD is a detection protocol that is designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. Because the network administrator can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different routing protocol hello mechanisms, network profiling and planning is easier, and reconvergence time is consistent and predictable. The main benefit of implementing BFD for BGP is a significantly faster reconvergence time. For internal BGP (iBGP) sessions and external BGP (eBGP) sessions that are either single hop or multihop, BGP can use of the multihop BFD support to help improve the BGP convergence because BFD detection and failure times are faster than the IGP convergence times in most of the network topologies. BGP needs the support of multihop BFD as described in RFC5882, *Generic Application of Bidirectional Forwarding Detection (BFD).*

BGP by default will purge the routes received from a specific peer when a BFD down event occurs and BFD informs BGP about it. The cBit in BFD determines whether BFD is dependent or independent of the Control Plane. Clients like BGP, whose peers are enabled with fast fall over feature with BFD support, can use this BFD cBit support to provide a more deterministic mechanism to do nonstop forwarding (NSF) when BGP graceful restart is enabled along with BFD fast-fallover support for BGP sessions.

When BGP is using BFD for the fast fallover feature for remote connectivity detection, BFD can detect some of those failures. If BFD is independent of the control plane, a BFD session failure means that data cannot be forwarded anymore (due to link control failures) and so the BGP graceful restart procedures should be terminated to avoid null routes. On the other hand, when BFD is dependent on the control plane, a BFD failure cannot be separated out from the other events taking place in the control plane. When the control plane crashes, a switchover happens and BFD restarts. It is best for the clients (like BGP) to avoid any terminations due to the graceful restart taking place.

The table below describes the handling of BFD down events by BGP.

Table 3: BGP handling of BFD Down Event

BFD Down Event	Failure—Control Plane Independent?	BGP Action for NSF (when GR and BFD are enabled)
BGP control plane failure detection enabled	Yes	Purge Routes
BGP control plane failure detection enabled	No	Carry on NSF and keep stale routes in Routing Information Base (RIB)
BGP control plane failure detection disabled (the default behavior)	Yes	Purge Routes
BGP control plane failure detection disabled (the default behavior)	No	Purge Routes

BGP session establishment works independently from BFD state change, except for fast fall-over detection, that is, inaccessible next-hop and cause best path re-calculation. This means that the BGP session could be established while BFD state is down or dampened, even with neighbor fail-over bfd configured.

From the XE 3.17S release the new optional keyword strict-mode is introduced, which does not allow BGP session to become established, if BFD is in down state. When BFD is dampened or down the routing protocol states or sessions cannot come up.

How to Configure BFD - BGP Multihop Client Support and cBit (IPv4 and IPv6)

Configuring BFD—BGP Multihop Client Support, cBit (IPv4 and IPv6), and Strict Mode

Before you begin

Note The multihop BFD minimum detection time should be higher than IGP convergence times in your network to ensure that down events are not mistakenly identified during reconvergences, causing multihop BGP sessions to flap.

Note For the BFD strict mode to work, configure BFD on both the neighboring devices.

Procedure

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 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 50000
```

Configures the Border Gateway Protocol (BGP) routing process and enters router configuration mode.

Step 4 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 10.0.0.2 remote-as 100
```

Adds an entry to the BGP or multiprotocol BGP neighbor table.

Step 5 **neighbor** *ip-address* **update-source** *interface-type interface-number*

Example:

```
Device(config-router)# neighbor 10.0.0.2 update-source GigabitEthernet 0/0/0
```

Allows BGP sessions to use any operational interface for TCP connections.

Step 6 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 10.0.0.2 remote-as 100
```

Adds an entry to the BGP or multiprotocol BGP neighbor table.

Step 7 **neighbor** *ip-address* **ebgp-multihop** *ttl*

Example:

```
Device(config-router)# neighbor 10.0.0.2 ebgp-multihop 4
```

Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.

Step 8 **neighbor** *ip-address* **fall-over** **bfd** [**multi-hop**] [**check-control-plane-failure**] [**strict-mode**]

Example:

```
Device(config-router)# neighbor 10.0.0.2 fall-over bfd multi-hop check-control-plane-failure
 strict-mode
```

- Enables BGP to monitor the peering session of a specified neighbor for adjacency changes and to deactivate the peering session.
- Configures BGP BFD with control plane independence that is enabled for BFD cBit support.

Note When **check-control-plane-failure** is enabled for the neighbor and the trigger for adjacency down is set at BFD-Down notification, then BGP looks into C-Bit to decide whether to carry out Cisco Nonstop Forwarding or Purge the routes. BGP looks into C-Bit in BFD packet and decides only when **check-control-plane-failure** is enabled.

Step 9 **end**

Example:

```
Device(config-router)# end
```

Exits router configuration mode and returns to privileged EXEC mode.

Configuration Examples for BFD - BGP Multihop Client Support and cBit (IPv4 and IPv6)

Example: Configuring BFD—BGP Multihop Client Support, cBit (IPv4/IPv6), and Strict Mode

```
R1 e0/0 ---------------e0/0 R2

Router 1 configuration

hostname R1
!
bfd map ipv4 10.2.2.2/32 10.1.1.1/32 mh1
!
bfd-template multi-hop mh1
interval min-tx 200 min-rx 200 multiplier 3
!
interface Loopback1
ip address 10.1.1.1 255.255.255.255
ip ospf 1 area 0
!
interface Ethernet0/0
ip address 10.0.0.1 255.255.255.0
ip ospf 1 area 0
!
router ospf 1
!
router bgp 1
neighbor 10.2.2.2 remote-as 1
neighbor 10.2.2.2 update-source Loopback1
neighbor 10.2.2.2 fall-over bfd multi-hop check-control-plane-failure strict-mode
!
address-family ipv4
neighbor 10.2.2.2 activate
exit-address-family
!

Router 2 configuration:

hostname R2
!
bfd map ipv4 10.1.1.1/32 10.2.2.2/32 mh1
bfd-template multi-hop mh1
interval min-tx 200 min-rx 200 multiplier 3
!
interface Loopback1
ip address 10.2.2.2 255.255.255.255
ip ospf 1 area 0
!
interface Ethernet0/0
```

```
ip address 10.0.0.2 255.255.255.0
ip ospf 1 area 0
!
router ospf 1
!
router bgp 1
neighbor 10.1.1.1 remote-as 1
neighbor 10.1.1.1 update-source Loopback1
neighbor 10.1.1.1 fall-over bfd multi-hop check-control-plane-failure strict-mode
!
address-family ipv4
neighbor 10.1.1.1 activate
exit-address-family
!
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference

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

CHAPTER 7

VPLS BGP Signaling

The two primary functions of the Virtual Private LAN Service (VPLS) control plane are autodiscovery and signaling. The VPLS BGP Signaling feature enables you to use BGP as both an autodiscovery and a signaling protocol for VPLS, in accordance with RFC 4761.

Prerequisites for VPLS BGP Signaling

You are familiar with the concepts in the “Configuring Virtual Private LAN Services” and the “VPLS Autodiscovery BGP Based” modules of the MPLS Layer 2 VPNs Configuration Guide .

Information About VPLS BGP Signaling

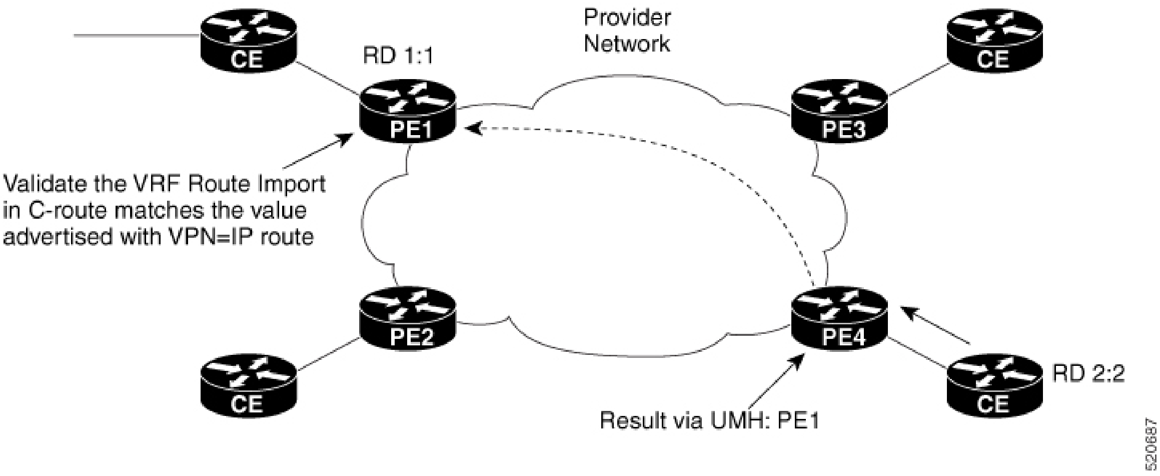
Overview of VPLS BGP Signaling

Prior to the VPLS BGP Signaling feature, BGP was used for autodiscovery and Label Distribution Protocol (LDP) for signaling in accordance with RFC 6074. The VPLS BGP Signaling feature enables you to use BGP as the control plane protocol for both autodiscovery and signaling in accordance with RFC 4761.

As specified in RFC 4761, internal BGP (iBGP) peers will exchange update messages of the L2VPN AFI/SAFI with L2VPN information to perform both autodiscovery and signaling. The BGP multiprotocol Network Layer Reachability Information (NLRI) consists of a Route Distinguisher (RD), VPLS Endpoint ID (VE ID), VE Block Offset (VBO), VE Block Size (VBS), and Label Base (LB).

The figure below shows the format of the NLRI for RFC 4761.

Figure 10: RFC 4761 NLRI


Provider
Network
RD 1:1
CE
PE1
PE3
CE
Validate the VRF Route Import
in C-route matches the value
advertised with VPN=IP route
PE2
PE4
CE
CE
RD 2:2
Result via UMH: PE1
520687

Additional information, such as next-hop, route target (specified for a VPLS instance), and other Layer 2 data are carried in the BGP extended community attributes. A route target-based import/export mechanism similar to L3VPN is performed by BGP to filter L2VPN NLRIs of a particular VPLS instance.

Whether you use BGP signaling (RFC 4761) or LDP signaling (RFC 6074) depends on the commands you specify. To enable the VPLS BGP Signaling feature, use the **autodiscovery bgp signaling bgp** command in L2 VFI configuration mode. This command is supported on a per VPLS instance basis.

If a BGP session receives an invalid (that is, not matching the configuration) BGP update advertisement (update or withdraw), it is ignored.

BGP's main task in supporting VPLS is route distribution via the L2VPN address family and interactions with L2VPN. Interactions between BGP and other components remain the same. Basic BGP functionalities like best-path selection, next-hop handling, and update generation, continue to operate in the same manner with VPLS BGP signaling. BGP RT constraint works seamlessly with the BGP VPLS Signaling feature.

The above example shows sample configuration on one PE. Similar configuration can be mirrored on other PEs.

How to Configure VPLS BGP Signaling

Configuring VPLS BGP Signaling

Before you begin

Note For more information, see *Configuring Virtual Private Lan Services*.

Procedure

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 **l2vpn vfi context** *name*

Example:

```
Device(config)# l2vpn vfi context vfi1
```

Establishes a L2VPN virtual forwarding interface (VFI) between two or more separate networks and enters Layer 2 VFI configuration mode.

Step 4 **vpn id** *vpn-id*

Example:

```
Device(config-vfi)# vpn id 100
```

Configures a VPN ID for the VPLS domain.

Step 5 **autodiscovery bgp signaling** {**bgp** | **ldp**} [**template** *template-name*]

Example:

```
Device(config-vfi)# autodiscovery bgp signaling bgp
```

Enables BGP signaling and discovery or LDP signaling and enters L2VPN VFI autodiscovery configuration mode.

Note For the VPLS BGP Signaling feature use the **autodiscovery bgp signaling bgp** command.

Step 6 **ve id** *ve-id*

Example:

```
Device(config-vfi-autodiscovery)# ve id 1001
```

Specifies the VPLS endpoint (VE) device ID value. The VE ID identifies a VFI within a VPLS service. The VE device ID value is from 1 to 16384.

Step 7 **ve range** *ve-range*

Example:

```
Device(config-vfi-autodiscovery)# ve range 12
```

Specifies the VE device ID range value. The VE range overrides the minimum size of VE blocks. The default minimum size is 10. Any configured VE range must be higher than 10.

Step 8 **exit**

Example:

```
Device(config-vfi-autodiscovery)# exit
```

Exits L2VPN VFI autodiscovery configuration mode and enters L2VPN VFI configuration mode.

Step 9 **exit**

Example:

```
Device(config-vfi)# exit
```

Exits L2VPN VFI configuration mode and enters global configuration mode.

Step 10 **router bgp** *autonomous-system-number*

Example:

```
Device(config)# router bgp 100
```

Enters router configuration mode to create or configure a BGP routing process.

Step 11 **bgp graceful-restart**

Example:

```
Device(config-router)# bgp graceful-restart
```

Enables the BGP graceful restart capability and BGP nonstop forwarding (NSF) awareness.

Step 12 **neighbor** *ip-address* **remote-as** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 10.10.10.1 remote-as 100
```

Configures peering with a BGP neighbor in the specified autonomous system.

Step 13 **address-family l2vpn** [**vpls**]

Example:

```
Device(config-router)# address-family l2vpn vpls
```

Specifies the L2VPN address family and enters address family configuration mode.

- The optional **vpls** keyword specifies that VPLS endpoint provisioning information is to be distributed to BGP peers.

In this example, an L2VPN VPLS address family session is created.

Step 14 **neighbor** *ip-address* **activate**

Example:

```
Device(config-router-af)# neighbor 10.10.10.1 activate
```

Enables the neighbor to exchange information for the L2VPN VPLS address family with the local device.

Step 15 **neighbor** *ip-address* **send-community** [**both** | **standard** | **extended**]

Example:

```
Device(config-router-af)# neighbor 10.10.10.1 send-community extended
```

Specifies that a communities attribute should be sent to a BGP neighbor.

- In this example, an extended communities attribute is sent to the neighbor at 10.10.10.1.

Step 16 **neighbor** *ip-address* **suppress-signaling-protocol ldp**

Example:

```
Device(config-router-af)# neighbor 10.10.10.1 suppress-signaling-protocol ldp
```

Suppresses LDP signaling and enables BGP signaling.

- In this example LDP signaling is suppressed (and BGP signaling enabled) for the neighbor at 10.10.10.1.

Step 17 **end**

Example:

```
Device(config-router-af)# end
```

Exits address family configuration mode and returns to privileged EXEC mode.

Step 18 **show bgp l2vpn vpls** {**all** | **rd** *route-distinguisher*}

Example:

```
Device# show bgp l2vpn vpls all
```

(Optional) Displays information about the L2VPN VPLS address family.

Configuration Examples for VPLS BGP Signaling

Example: Configuring and Verifying VPLS BGP Signaling

```
l2vpn vfi context vfi1
 vpn id 100
 autodiscovery bgp signaling bgp
  ve id 1001
  ve range 10
 !
!
router bgp 100
 bgp graceful-restart
 neighbor 192.168.200.224 remote-as 100
 neighbor 192.168.200.224 update-source Loopback1
 !
 address-family l2vpn vpls
  neighbor 192.168.200.224 activate
  neighbor 192.168.200.224 send-community extended
  neighbor 192.168.200.224 suppress-signaling-protocol ldp
  exit-address-family
 !
show bgp l2vpn vpls all

Network                            Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:100
*>100:100:VEID-1001:Blk-1001/136    10.0.0.0                              32768  ?

*>i 100:100:VEID-1003:Blk-1000/136  192.168.200.224               0      100     0      ?
```

CHAPTER 8

BGP Support for 4-byte ASN

The Cisco implementation of 4-byte autonomous system (AS) numbers uses asplain (65538, for example) as the default regular expression match and the output display format for AS numbers. However, you can configure 4-byte AS numbers in both the asplain format and the asdot format as described in RFC 5396. In addition, 4-byte ASN route distinguisher (RD) and route target (RT) BGP support for 4-byte autonomous numbers is added.

Information About BGP Support for 4-byte ASN

BGP Autonomous System Number Formats

Prior to January 2009, BGP autonomous system numbers that were allocated to companies were 2-octet numbers in the range from 1 to 65535 as described in RFC 4271, *A Border Gateway Protocol 4 (BGP-4)*. Due to increased demand for autonomous system numbers, the Internet Assigned Number Authority (IANA) started in January 2009 will allocate four-octet autonomous system numbers in the range from 65536 to 4294967295. RFC 5396, *Textual Representation of Autonomous System (AS) Numbers*, documents three methods of representing autonomous system numbers. Cisco has implemented the following two methods:

- Asplain—Decimal value notation where both 2-byte and 4-byte autonomous system numbers are represented by their decimal value. For example, 65526 is a 2-byte autonomous system number and 234567 is a 4-byte autonomous system number.

- Asdot—Autonomous system dot notation where 2-byte autonomous system numbers are represented by their decimal value and 4-byte autonomous system numbers are represented by a dot notation. For example, 65526 is a 2-byte autonomous system number and 1.169031 is a 4-byte autonomous system number (this is dot notation for the 234567 decimal number).

For details about the third method of representing autonomous system numbers, see RFC 5396.

Asdot Only Autonomous System Number Formatting

In Cisco IOS XE Release 2.3, the 4-octet (4-byte) autonomous system numbers are entered and displayed only in asdot notation, for example, 1.10 or 45000.64000. When using regular expressions to match 4-byte autonomous system numbers the asdot format includes a period, which is a special character in regular expressions. A backslash must be entered before the period (for example, 1\.14) to ensure the regular expression match does not fail. The table below shows the format in which 2-byte and 4-byte autonomous system numbers are configured, matched in regular expressions, and displayed in **show** command output in Cisco IOS images where only asdot formatting is available.

Table 4: Asdot Only 4-Byte Autonomous System Number Format

Format	Configuration Format	Show Command Output and Regular Expression Match Format
asdot	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535

Asplain as Default Autonomous System Number Formatting

In Cisco IOS XE Release 2.4 and later releases, the Cisco implementation of 4-byte autonomous system numbers uses asplain as the default display format for autonomous system numbers, but you can configure 4-byte autonomous system numbers in both the asplain and asdot format. In addition, the default format for matching 4-byte autonomous system numbers in regular expressions is asplain, so you must ensure that any regular expressions to match 4-byte autonomous system numbers are written in the asplain format. If you want to change the default **show** command output to display 4-byte autonomous system numbers in the asdot format, use the **bgp asnotation dot** command under router configuration mode. When the asdot format is enabled as the default, any regular expressions to match 4-byte autonomous system numbers must be written using the asdot format, or the regular expression match will fail. The tables below show that although you can configure 4-byte autonomous system numbers in either asplain or asdot format, only one format is used to display **show** command output and control 4-byte autonomous system number matching for regular expressions, and the default is asplain format. To display 4-byte autonomous system numbers in **show** command output and to control matching for regular expressions in the asdot format, you must configure the **bgp asnotation dot** command. After enabling the **bgp asnotation dot** command, a hard reset must be initiated for all BGP sessions by entering the **clear ip bgp *** command.

Note If you are upgrading to an image that supports 4-byte autonomous system numbers, you can still use 2-byte autonomous system numbers. The **show** command output and regular expression match are not changed and remain in asplain (decimal value) format for 2-byte autonomous system numbers regardless of the format configured for 4-byte autonomous system numbers.

Table 5: Default Asplain 4-Byte Autonomous System Number Format

Format	Configuration Format	Show Command Output and Regular Expression Match Format
asplain	2-byte: 1 to 65535 4-byte: 65536 to 4294967295	2-byte: 1 to 65535 4-byte: 65536 to 4294967295
asdot	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535	2-byte: 1 to 65535 4-byte: 65536 to 4294967295

Table 6: Asdot 4-Byte Autonomous System Number Format

Format	Configuration Format	Show Command Output and Regular Expression Match Format
asplain	2-byte: 1 to 65535 4-byte: 65536 to 4294967295	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535
asdot	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535	2-byte: 1 to 65535 4-byte: 1.0 to 65535.65535

Reserved and Private Autonomous System Numbers

In Cisco IOS XE Release 2.3 and later releases, the Cisco implementation of BGP supports RFC 4893. RFC 4893 was developed to allow BGP to support a gradual transition from 2-byte autonomous system numbers to 4-byte autonomous system numbers. A new reserved (private) autonomous system number, 23456, was created by RFC 4893 and this number cannot be configured as an autonomous system number in the Cisco IOS CLI.

RFC 5398, *Autonomous System (AS) Number Reservation for Documentation Use*, describes new reserved autonomous system numbers for documentation purposes. Use of the reserved numbers allow configuration examples to be accurately documented and avoids conflict with production networks if these configurations are literally copied. The reserved numbers are documented in the IANA autonomous system number registry. Reserved 2-byte autonomous system numbers are in the contiguous block, 64496 to 64511 and reserved 4-byte autonomous system numbers are from 65536 to 65551 inclusive.

Private 2-byte autonomous system numbers are still valid in the range from 64512 to 65534 with 65535 being reserved for special use. Private autonomous system numbers can be used for internal routing domains but must be translated for traffic that is routed out to the Internet. BGP should not be configured to advertise private autonomous system numbers to external networks. Cisco IOS software does not remove private autonomous system numbers from routing updates by default. We recommend that ISPs filter private autonomous system numbers.

Note Autonomous system number assignment for public and private networks is governed by the IANA. For information about autonomous-system numbers, including reserved number assignment, or to apply to register an autonomous system number, see the following URL: http://www.iana.org/.

Cisco Implementation of 4-Byte Autonomous System Numbers

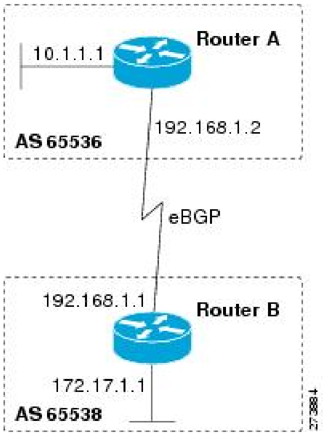
In Cisco IOS XE Release 2.4 and later releases, the Cisco implementation of 4-byte autonomous system (AS) numbers uses asplain—65538, for example—as the default regular expression match and output display format for AS numbers, but you can configure 4-byte AS numbers in both the asplain format and the asdot format as described in RFC 5396. To change the default regular expression match and output display of 4-byte AS numbers to asdot format, use the **bgp asnotation dot** command followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions. For more details about 4-byte AS number formats, see the “BGP Autonomous System Number Formats” section.

In Cisco IOS XE Release 2.3, the Cisco implementation of 4-byte AS numbers uses asdot—1.2, for example—as the only configuration format, regular expression match, and output display, with no asplain support. For an example of BGP peers in two autonomous systems using 4-byte numbers, see the figure below. To view a configuration example of the configuration between three neighbor peers in separate 4-byte autonomous systems configured using asdot notation, see the "Example: Configuring a BGP Routing Process and Peers Using 4-Byte Autonomous System Numbers" section.

Cisco also supports RFC 4893, which was developed to allow BGP to support a gradual transition from 2-byte AS numbers to 4-byte AS numbers. To ensure a smooth transition, we recommend that all BGP speakers within an AS that is identified using a 4-byte AS number be upgraded to support 4-byte AS numbers.

Note A new private AS number, 23456, was created by RFC 4893, and this number cannot be configured as an AS number in the Cisco IOS CLI.

Figure 11: BGP Peers in Two Autonomous Systems Using 4-Byte Numbers



How to Configure BGP Support for 4-byte ASN

Configuring a BGP Routing Process and Peers Using 4-Byte Autonomous System Numbers

Perform this task to configure a Border Gateway Protocol (BGP) routing process and BGP peers when the BGP peers are located in an autonomous system (AS) that uses 4-byte AS numbers. The address family configured here is the default IPv4 unicast address family, and the configuration is done at Router B in the figure above (in the "Cisco Implementation of 4-Byte Autonomous System Numbers" section). The 4-byte AS numbers in this task are formatted in the default asplain (decimal value) format; for example, Router B is in AS number 65538 in the figure above. Remember to perform this task for any neighbor routers that are to be BGP peers.

Before you begin

Note By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, such as IPv6 prefixes, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types.

Procedure

	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	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 65536`	Adds the IP address of the neighbor in the specified AS to the IPv4 multiprotocol BGP neighbor table of the local device. • In this example, the 4-byte AS number, 65536, is defined in asplain notation.
Step 5	Repeat Step 4 to define other BGP neighbors, as required.	--
Step 6	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes.

	Command or Action	Purpose
		• The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.
Step 7	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local device.
Step 8	Repeat Step 7 to activate other BGP neighbors, as required.	--
Step 9	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this AS and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.

Step 10	**end**
	Example:
	`Device(config-router-af)# end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Step 11	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp 10.1.1.0`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 12	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	(Optional) Displays the status of all BGP connections.

Examples

The following output from the **show ip bgp** command at Router B shows the BGP routing table entry for network 10.1.1.0 learned from the BGP neighbor at 192.168.1.2 in Router A in the figure above with its 4-byte AS number of 65536 displayed in the default asplain format.

```
RouterB# show ip bgp 10.1.1.0

BGP routing table entry for 10.1.1.0/24, version 2
Paths: (1 available, best #1)
  Advertised to update-groups:
     2
  65536
    192.168.1.2 from 192.168.1.2 (10.1.1.99)
      Origin IGP, metric 0, localpref 100, valid, external, best
```

The following output from the **show ip bgp summary** command shows the 4-byte AS number 65536 for the BGP neighbor 192.168.1.2 of Router A in the figure above after this task has been configured on Router B:

```
RouterB# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 65538
BGP table version is 3, main routing table version 3
2 network entries using 234 bytes of memory
2 path entries using 104 bytes of memory
3/2 BGP path/bestpath attribute entries using 444 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 806 total bytes of memory
BGP activity 2/0 prefixes, 2/0 paths, scan interval 60 secs
Neighbor        V          AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down  Stated
192.168.1.2     4       65536       6       6        3    0    0 00:01:33        1
```

Troubleshooting Tips

Use the **ping** command to verify basic network connectivity between the BGP devices.

Modifying the Default Output and Regular Expression Match Format for 4-Byte Autonomous System Numbers

Perform this task to modify the default output format for 4-byte autonomous system (AS) numbers from asplain format to asdot notation format. The **show ip bgp summary** command is used to display the changes in output format for the 4-byte AS numbers.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays the status of all Border Gateway Protocol (BGP) connections.

	Command or Action	Purpose
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. - In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 5	**bgp asnotation dot** **Example:** `Device(config-router)# bgp asnotation dot`	Changes the default output format of BGP 4-byte AS numbers from asplain (decimal values) to dot notation. **Note** 4-byte AS numbers can be configured using either asplain format or asdot format. This command affects only the output displayed for **show** commands or the matching of regular expressions.

Step 6	**end**
	Example:
	`Device(config-router)# end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Step 7	**clear ip bgp *** **Example:** `Device# clear ip bgp *`	Clears and resets all current BGP sessions. - In this example, a hard reset is performed to ensure that the 4-byte AS number format change is reflected in all BGP sessions. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 8	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays the status of all BGP connections.
Step 9	**show ip bgp regexp** *regexp* **Example:**	Displays routes that match the AS path regular expression.

	Command or Action	Purpose
	`Device# show ip bgp regexp ^1\.0$`	- In this example, a regular expression to match a 4-byte AS path is configured using asdot format.
Step 10	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 11	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. - In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 12	**no bgp asnotation dot** **Example:** `Device(config-router)# no bgp asnotation dot`	Resets the default output format of BGP 4-byte AS numbers back to asplain (decimal values). **Note** 4-byte AS numbers can be configured using either asplain format or asdot format. This command affects only the output displayed for **show** commands or the matching of regular expressions.

Step 13	**end**
	Example:
	`Device(config-router)# end`
		Exits router configuration mode and returns to privileged EXEC mode.

Step 14	**clear ip bgp *** **Example:** `Device# clear ip bgp *`	Clears and resets all current BGP sessions. - In this example, a hard reset is performed to ensure that the 4-byte AS number format change is reflected in all BGP sessions. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following output from the **show ip bgp summary** command shows the default asplain format of the 4-byte AS numbers. Note the asplain format of the 4-byte AS numbers, 65536 and 65550.

```
Router# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 65538
BGP table version is 1, main routing table version 1
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  Statd
192.168.1.2     4       65536       7       7        1    0    0 00:03:04      0
192.168.3.2     4       65550       4       4        1    0    0 00:00:15      0
```

After the **bgp asnotation dot** command is configured (followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions), the output is converted to asdot notation format as shown in the following output from the **show ip bgp summary** command. Note the asdot format of the 4-byte AS numbers, 1.0 and 1.14 (these are the asdot conversions of the 65536 and 65550 AS numbers.

```
Router# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 1.2
BGP table version is 1, main routing table version 1
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  Statd
192.168.1.2     4         1.0       9       9        1    0    0 00:04:13      0
192.168.3.2     4        1.14       6       6        1    0    0 00:01:24      0
```

After the **bgp asnotation dot** command is configured (followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions), the regular expression match format for 4-byte AS paths is changed to asdot notation format. Although a 4-byte AS number can be configured in a regular expression using either asplain format or asdot format, only 4-byte AS numbers configured using the current default format are matched. In the first example below, the **show ip bgp regexp** command is configured with a 4-byte AS number in asplain format. The match fails because the default format is currently asdot format and there is no output. In the second example using asdot format, the match passes and the information about the 4-byte AS path is shown using the asdot notation.

Note The asdot notation uses a period, which is a special character in Cisco regular expressions. To remove the special meaning, use a backslash before the period.

```
Router# show ip bgp regexp ^65536$

Router# show ip bgp regexp ^1\.0$

BGP table version is 2, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 1.0 i
```

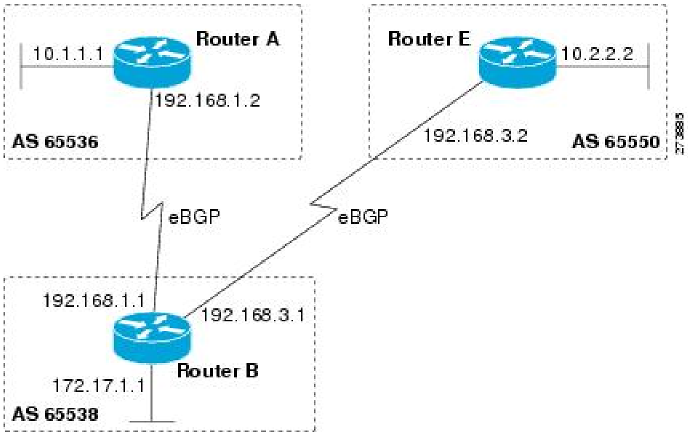
Configuration Examples for BGP Support for 4-byte ASN

Examples: Configuring a BGP Routing Process and Peers Using 4-Byte Autonomous System Numbers

Asplain Format

The following example shows the configuration for Router A, Router B, and Router E in the figure below with a Border Gateway Protocol (BGP) process configured between three neighbor peers (at Router A, at Router B, and at Router E) in separate 4-byte autonomous systems configured using asplain notation. IPv4 unicast routes are exchanged with all peers.

Figure 12: BGP Peers Using 4-Byte Autonomous System Numbers in Asplain Format



Router A

```
router bgp 65536
 bgp router-id 10.1.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.1 remote-as 65538
 !
 address-family ipv4
  neighbor 192.168.1.1 activate
  no auto-summary
  no synchronization
  network 10.1.1.0 mask 255.255.255.0
  exit-address-family
```

Router B

```
router bgp 65538
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 65536
 neighbor 192.168.3.2 remote-as 65550
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

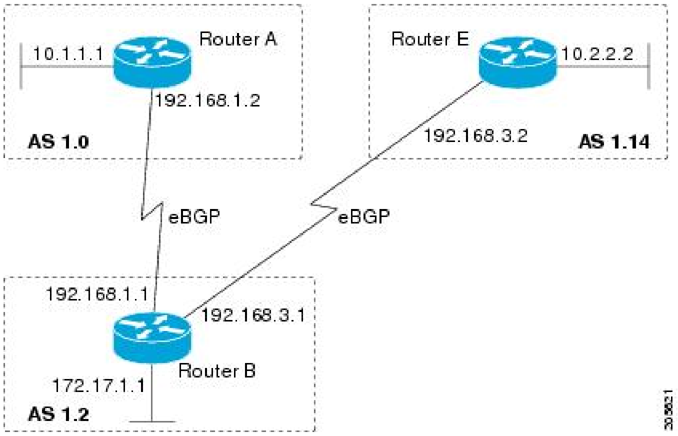
Router E

```
router bgp 65550
 bgp router-id 10.2.2.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.3.1 remote-as 65538
 !
 address-family ipv4
  neighbor 192.168.3.1 activate
  no auto-summary
  no synchronization
  network 10.2.2.0 mask 255.255.255.0
  exit-address-family
```

Asdot Format

The following example shows how to create the configuration for Router A, Router B, and Router E in the figure below with a BGP process configured between three neighbor peers (at Router A, at Router B, and at Router E) in separate 4-byte autonomous systems configured using the default asdot format. IPv4 unicast routes are exchanged with all peers.

Figure 13: BGP Peers Using 4-Byte Autonomous System Numbers in Asdot Format



Router A

```
router bgp 1.0
 bgp router-id 10.1.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.1 remote-as 1.2
 !
 address-family ipv4
  neighbor 192.168.1.1 activate
  no auto-summary
  no synchronization
  network 10.1.1.0 mask 255.255.255.0
  exit-address-family
```

Router B

```
router bgp 1.2
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 1.0
 neighbor 192.168.3.2 remote-as 1.14
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

Router E

```
router bgp 1.14
 bgp router-id 10.2.2.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.3.1 remote-as 1.2
 !
 address-family ipv4
  neighbor 192.168.3.1 activate
  no auto-summary
  no synchronization
  network 10.2.2.0 mask 255.255.255.0
  exit-address-family
```

Examples: Configuring a VRF and Setting an Extended Community Using a BGP 4-Byte Autonomous System Number

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example is available in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases and shows how to create a VRF with a route target that uses a 4-byte autonomous system number, 65537, and how to set the route target to extended community value 65537:100 for routes that are permitted by the route map:

```
ip vrf vpn_red
 rd 64500:100
 route-target both 65537:100
 exit
route-map red_map permit 10
 set extcommunity rt 65537:100
 end
```

After the configuration is completed, use the **show route-map** command to verify that the extended community is set to the route target that contains the 4-byte autonomous system number of 65537:

```
RouterB# show route-map red_map
route-map red_map, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:65537:100
  Policy routing matches: 0 packets, 0 bytes
```

4-Byte Autonomous System Number RD Support

The following example shows how to create a VRF with a route distinguisher that contains a 4-byte AS number 65536, and a route target that contains a 4-byte autonomous system number, 65537:

```
ip vrf vpn_red
 rd 65536:100
 route-target both 65537:100
 exit
```

After the configuration is completed, use the **show vrf** command to verify that the 4-byte AS number route distinguisher is set to 65536:100:

```
RouterB# show vrf vpn_red
Current configuration : 36 bytes
vrf definition x
 rd 65536:100
!
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T

The following example is available in Cisco IOS Release 12.0(32)S12, and 12.4(24)T and shows how to create a VRF with a route target that uses a 4-byte autonomous system number, 1.1, and how to set the route target to the extended community value 1.1:100 for routes that are permitted by the route map.

Note In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, and later releases, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

```
ip vrf vpn_red
 rd 64500:100
 route-target both 1.1:100
 exit
route-map red_map permit 10
 set extcommunity rt 1.1:100
 end
```

After the configuration is completed, use the **show route-map** command to verify that the extended community is set to the route target that contains the 4-byte autonomous system number of 1.1.

```
RouterB# show route-map red_map
route-map red_map, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:1.1:100
  Policy routing matches: 0 packets, 0 bytes
```

Asdot Default Format for 4-Byte Autonomous System Number RD Support

The following example works if you have configured asdot as the default display format using the **bgp asnotation dot** command:

```
ip vrf vpn_red
 rd 1.0:100
 route-target both 1.1:100
 exit
```

Additional References for BGP Support for 4-byte ASN

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference

Standards and RFCs

Standard/RFC	Title
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5396	*Textual Representation of Autonomous System (AS) Numbers*
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*

Standard/RFC	Title
RFC 5668	*4-Octet AS Specific BGP Extended Community*

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

CHAPTER 9

Labeled BGP Support

This feature module describes how to add label mapping information to the Border Gateway Protocol (BGP) message that is used to distribute the route on the router.

Prerequisites for Labeled BGP Support

Starting Cisco IOS XE Release 3.6.0S or later that supports Labeled BGP must be installed previously on the Cisco ASR 903 Series Aggregation Services Router.

Restrictions for Labeled BGP Support

- The router supports only the client functionality of RFC 3107 and not its area border router (ABR) functionality.
- The router does not support two label-pop (Label pop is the process of removing label header).

Overview of Labeled BGP Support

The Labeled BGP Support feature provides the option to use the BGP update message (that is used to distribute the route) to re-distribute Multiprotocol Label Switching (MPLS) label mapped to that route. The label mapping information is added (using send-label option of RFC 3107) to the same BGP message that is used to distribute the route. This process is useful in inter-domain routing, and the message that is used to distribute the route. This process is useful in inter-domain routing, and the router supports this functionality as well as the virtual private network (VPN) and virtual routing and forwarding (VRF) over Labeled BGP functionality.

VPN/VRF over RFC 3107

The VPN/VRF over Labeled BGP is a 3-label imposition process (VRF Label, BGP label, interior gateway protocols [IGP] label). The innermost label is VRF, followed by BGP (for RFC 3107), and IGP. This functionality allows the router to support a VRF over labeled BGP session with an ABR.

Configuring Labeled BGP Support

Procedure

	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	router bgp *as-number* **Example:** `Router(config)# router bgp 100`	Enters router configuration mode. • as-number—Number of an autonomous system that identifies the router to other BGP routers and tags the routing information that is passed along. The valid values range from 1 to 65535.
Step 4	address family ipv4 **Example:** `Router(config-router)# address family ipv4`	Configures the address family as IPv4 using standard IPv4 address prefixes.
Step 5	neighbor peer-group-name send-community **Example:** `Router(config-router)# neighbor 172.16.70.23 send-community`	Specifies that the communities attribute be sent to the neighbor at this IP address. • peer-group-name—Name of a BGP peer group.
Step 6	neighbor peer-group-name next-hop-self **Example:** `Router(config-router)# neighbor 172.16.70.23 next-hop-self`	Configures the router as the next hop for a BGP-speaking neighbor or peer group.
Step 7	neighbor peer-group-name activate **Example:** `Router(config-router)# neighbor 172.16.70.23 activate`	Enables the exchange of information with a neighboring BGP router.

Configuration Example for Labeled BGP Support

>

The following is a sample configuration of the Labeled BGP Support feature.

```
!
router bgp 1000
 bgp router-id 100.111.13.23
 neighbor pan peer-group
 neighbor pan remote-as 1000
 neighbor pan update-source Loopback0
 neighbor 100.111.14.3 peer-group pan
 !
 address-family ipv4
  neighbor pan send-community
  neighbor pan next-hop-self
  neighbor pan send-label
!The send-label option is used to associate a BGP label to the prefix.
  neighbor 100.111.14.3 activate
 exit-address-family
 !
 address-family vpnv4
  neighbor pan send-community extended
  neighbor 100.111.14.3 activate
 exit-address-family
 !
 address-family ipv4 vrf LTE12
  redistribute connected
 exit-address-family
!
```

Verifying Labeled BGP Support

To verify the Labeled BGP Support, use the **show** commands given below:

```
Router# show bgp ipv4 unicast labels
Network          Next Hop      In label/Out label
  1.0.0.0          0.0.0.0         imp-null/nolabel
  10.13.22.2/31    0.0.0.0         imp-null/nolabel
  10.13.23.0/31    0.0.0.0         imp-null/nolabel
  10.70.1.0/30     0.0.0.0         imp-null/nolabel
  100.100.10.1/32  100.111.14.4    nolabel/558
                   100.111.14.3    nolabel/560
  100.100.13.23/32 0.0.0.0         imp-null/nolabel
  100.101.13.23/32 0.0.0.0         imp-null/nolabel
  100.111.13.23/32 0.0.0.0         imp-null/nolabel
  100.111.13.26/32 100.111.14.3    nolabel/534
                   100.111.14.4    nolabel/68
  100.111.15.1/32  100.111.14.3    nolabel/25
Router# show ip bgp labels
   Network          Next Hop      In label/Out label
   1.0.0.0          0.0.0.0         imp-null/nolabel
   10.13.22.2/31    0.0.0.0         imp-null/nolabel
   10.13.23.0/31    0.0.0.0         imp-null/nolabel
   10.70.1.0/30     0.0.0.0         imp-null/nolabel
   100.100.10.1/32  100.111.14.4    nolabel/563
                    100.111.14.3    nolabel/556
   100.100.13.23/32 0.0.0.0         imp-null/nolabel
   100.101.13.23/32 0.0.0.0         imp-null/nolabel
   100.111.13.23/32 0.0.0.0         imp-null/nolabel
   100.111.13.26/32 100.111.14.4    nolabel/561
                    100.111.14.3    nolabel/559
   100.111.15.1/32  100.111.14.4    nolabel/59
                    100.111.14.3    nolabel/57
   100.111.15.2/32  100.111.14.4    nolabel/62
                    100.111.14.3    nolabel/52
   100.112.1.1/32   100.111.14.4    nolabel/nolabel
```

```
                    100.111.14.3    nolabel/nolabel

   100.112.1.2/32   100.111.14.4    nolabel/nolabel

                    100.111.14.3    nolabel/nolabel

   100.112.1.3/32   100.111.14.4    nolabel/nolabel

                    100.111.14.3    nolabel/nolabel

Router# show ip bgp vpnv4 all label

   Network          Next Hop      In label/Out label

Route Distinguisher: 236:236

   154.154.236.4/30 100.154.1.1     nolabel/14002

                    100.154.1.1     nolabel/14002

   154.154.236.8/30 100.154.1.1     nolabel/14002

                    100.154.1.1     nolabel/14002

   154.154.236.12/30

                    100.154.1.1     nolabel/14002

                    100.154.1.1     nolabel/14002

   154.154.236.16/30

                    100.154.1.1     nolabel/14002

                    100.154.1.1     nolabel/14002

   154.154.236.20/30

                    100.154.1.1     nolabel/14002

                    100.154.1.1     nolabel/14002

   154.154.236.24/30

                    100.154.1.1     nolabel/14002

                    100.154.1.1     nolabel/14002

Router# show ip vrf interface

Interface              IP-Address      VRF                              Protocol

Vl100                  113.23.12.1     LTE12

Router# show ip bgp vpnv4 vrf LTE12 label

   Network          Next Hop      In label/Out label

Route Distinguisher: 6666:6666 (LTE12)

   113.22.12.0/24   100.111.13.22   nolabel/51

                    100.111.13.22   nolabel/51
   113.23.12.0/24   0.0.0.0         50/nolabel(LTE12)

   113.24.12.0/24   100.111.13.24   nolabel/32

                    100.111.13.24   nolabel/32

   115.1.12.0/24    100.111.15.1    nolabel/16024

                    100.111.15.1    nolabel/16024

   154.154.236.4/30 100.154.1.1     nolabel/14002

   154.154.236.8/30 100.154.1.1     nolabel/14002

   154.154.236.12/30

                    100.154.1.1     nolabel/14002

   154.154.236.16/30

                    100.154.1.1     nolabel/14002

   154.154.236.20/30

                    100.154.1.1     nolabel/14002

   154.154.236.24/30

                    100.154.1.1     nolabel/14002
```

To verify three Label Support, use the **show ip cef vrf** command as shown in the following example.

```
Router# show ip cef vrf LTE12 113.22.12.0 internal

113.22.12.0/24, epoch 0, flags rib defined all labels, RIB[B], refcount 5, per-destination
 sharing

  sources: RIB

  feature space:

   IPRM: 0x00018000

   LFD: 113.22.12.0/24  0 local labels

        contains path extension list

  ifnums: (none)

  path 13E8A064, path list 13F49DC8, share 1/1, type recursive, for IPv4, flags
must-be-labelled, recursive-via-host

    MPLS short path extensions: MOI flags = 0x0 label 51

  recursive via 100.111.13.22[IPv4:Default] label 51, fib 141253D8, 1 terminal fib,
v4:Default:100.111.13.22/32

    path 12520C8C, path list 13F49C38, share 1/1, type attached nexthop, for IPv4

   MPLS short path extensions: MOI flags = 0x0 label 17

      nexthop 100.111.14.4 Vlan10 label 17, adjacency IP adj out of Vlan10, addr 10.13.23.1
 13734C80

  output chain: label 22 label 51 label 17 TAG adj out of Vlan10, addr 10.13.23.1 143EDCA0
```

```
!You can see three labels in the output chain; of which 22 is VRF label, 51 is BGP label
!and 17 is LDP label
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS XE Command Reference	Cisco IOS IP Routing: BGP Command Reference

Standards and RFCs

Standard/RFC	Title
RFC-3107	Carrying Label Information in BGP-4

MIBs

MIB	MIBs Link
NA	To locate and download MIBs for selected platforms, Cisco IOS 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 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

CHAPTER 10

BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

The BGP Multipath Load Sharing for both eBGP and iBGP in an MPLS-VPN feature allows you to configure multipath load balancing with both external BGP (eBGP) and internal BGP (iBGP) paths in Border Gateway Protocol (BGP) networks that are configured to use Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs). This feature provides improved load balancing deployment and service offering capabilities and is useful for multihomed autonomous systems and Provider Edge (PE) routers that import both eBGP and iBGP paths from multihomed and stub networks.

Prerequisites for BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Load Balancing is Configured Under CEF

Cisco Express Forwarding (CEF) or distributed CEF (dCEF) must be enabled on all participating routers.

PIC Core should be Enabled

To enable load balancing over BGP multi-paths:

- Make sure BGP PIC core is enabled on the Cisco RSP3.

Note By default PIC core is enabled on the Cisco RSP3. You can also enable PIC core using the following global command: cef table output-chain build favor convergence-speed

- Advertise a minimum of 16 Global prefix scale on remote BGP peer.

Restrictions for BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Address Family Support

This feature is configured on a per VPN routing and forwarding instance (VRF) basis. This feature can be configured under only the IPv4 VRF address family.

Memory Consumption Restriction

Each BGP multipath routing table entry will use additional memory. We recommend that you do not use this feature on a router with a low amount of available memory and especially if router is carries full Internet routing tables.

Route Reflector Limitation

When multiple iBGP paths installed in a routing table, a route reflector will advertise only one paths (next hop). If a router is behind a route reflector, all routers that are connected to multihomed sites will not be advertised unless a different route distinguisher is configured for each VRF.

Restrictions for BGP Multipath Load Sharing on Cisco RSP3

- This feature can be configured under IPv4 address-family and IPv4 VRF address family.
- BGP maximum path is not supported over MPLS TE and MPLS TE FRR.
- Hierarchical BGP PIC edge or BGP maximum paths is not supported.
- The combination of IPv4 address family PIC edge with VPNv4 PIC edge is not supported.
- The combination of IPv4 address family maximum paths with VPNv4 maximum paths is not supported.
- L2VPN over BGP Multipath is not supported.

Information About BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Multipath Load Sharing Between eBGP and iBGP

A BGP routing process will install a single path as the best path in the routing information base (RIB) by default. The **maximum-paths** command allows you to configure BGP to install multiple paths in the RIB for multipath load sharing. BGP uses the best path algorithm to still select a single multipath as the best path and advertise the best path to BGP peers.

Note The number of paths of multipaths that can be configured is documented on the **maximum-paths** command reference page.

Load balancing over the multipaths is performed by CEF. CEF load balancing is configured on a per-packet round robin or on a per session (source and destination pair) basis. For information about CEF, refer to the "Cisco Express Forwarding Overview" documentation:

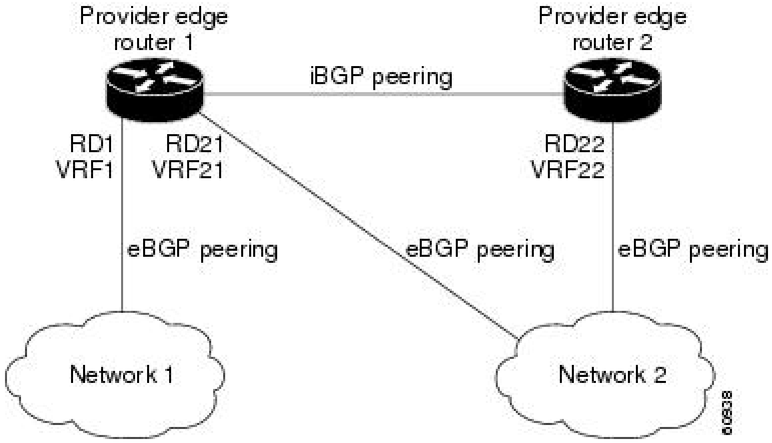
The BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature is enabled only under the IPv4 VRF address family configuration mode. When enabled, this feature can perform load balancing on eBGP and/or iBGP paths that are imported into the VRF. The number of multipaths is configured on a per VRF basis. Separate VRF multipath configurations are isolated by unique route distinguisher.

Note The BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature operates within the parameters of configured outbound routing policy.

eBGP and iBGP Multipath Load Sharing in a BGP MPLS Network

The figure below shows a service provider BGP MPLS network that connects two remote networks to PE router 1 and PE router 2. PE router 1 and PE router 2 are both configured for VPNv4 unicast iBGP peering. Network 2 is a multihomed network that is connected to PE router 1 and PE router 2. Network 2 also has extranet VPN services configured with Network 1. Both Network 1 and Network 2 are configured for eBGP peering with the PE routers.

Figure 14: A Service Provider BGP MPLS Network

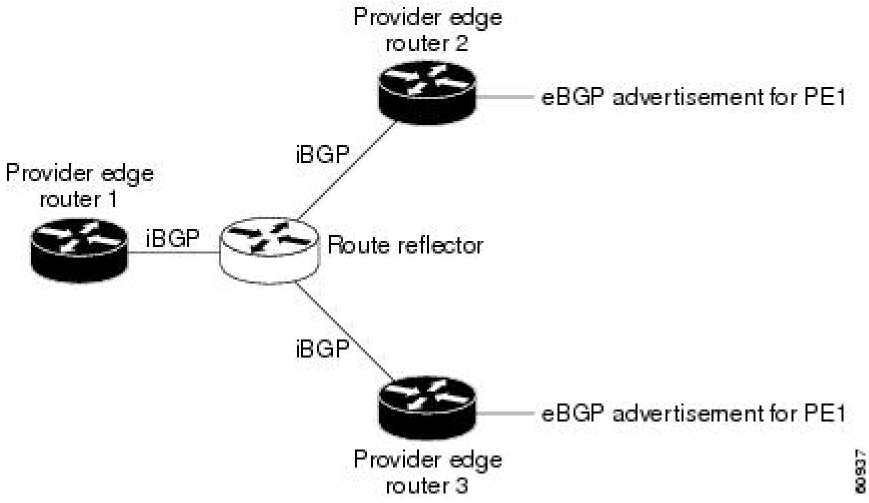


PE router 1 can be configured with the BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature so that both iBGP and eBGP paths can be selected as multipaths and imported into the VRF of Network 1. The multipaths will be used by CEF to perform load balancing. IP traffic that is sent from Network 2 to PE router 1 and PE router 2 will be sent across the eBGP paths as IP traffic. IP traffic that is sent across the iBGP path will be sent as MPLS traffic, and MPLS traffic that is sent across an eBGP path will be sent as IP traffic. Any prefix that is advertised from Network 2 will be received by PE router 1 through route distinguisher (RD) 21 and RD 22.The advertisement through RD 21 will be carried in IP packets, and the advertisement through RD 22 will be carried in MPLS packets. Both paths can be selected as multipaths for VRF1 and installed into the VRF1 RIB.

eBGP and iBGP Multipath Load Sharing With Route Reflectors

The figure below shows a topology that contains three PE routers and a route reflector, all configured for iBGP peering. PE router 2 and PE router 3 each advertise an equal preference eBGP path to PE router 1. By default, the route reflector will choose only one path and advertise PE router 1.

Figure 15: A Topology with a Route Reflector



For all equal preference paths to PE router 1 to be advertised through the route reflector, you must configure each VRF with a different RD. The prefixes received by the route reflector will be recognized differently and advertised to PE router 1.

Benefits of Multipath Load Sharing for Both eBGP and iBGP

The BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature allows multihomed autonomous systems and PE routers to be configured to distribute traffic across both eBGP and iBGP paths.

How to Configure BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Configuring Multipath Load Sharing for Both eBGP and iBGP

To configure this feature, perform the steps in this section.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**address-family ipv4 vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf RED`	Places the router in address family configuration mode. • Separate VRF multipath configurations are isolated by unique route distinguisher.
Step 5	**maximum-paths eibgp** *number* **Example:** `Device(config-router-af)# maximum-paths eibgp 6`	Configures the number of parallel iBGP and eBGP routes that can be installed into a routing table.

	Command or Action	Purpose
		Note The **maximum-paths eibgp** command can be configured only under the IPv4 VRF address family configuration mode and cannot be configured in any other address family configuration mode.

Step 6	**end**
	Example:
	`Device(config-router-af)# end`
		Exits address family configuration mode, and enters privileged EXEC mode.

Verifying Multipath Load Sharing for Both eBGP an iBGP

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp neighbors** [*neighbor-address* [**advertised-routes** | **dampened-routes** | **flap-statistics** | **paths** [*regexp*] | **received** *prefix-filter* | **received-routes** | **routes**]] **Example:** `Device# show ip bgp neighbors`	Displays information about the TCP and BGP connections to neighbors.
Step 3	**show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} **Example:** `Device# show ip bgp vpnv4 vrf RED`	Displays VPN address information from the BGP table. This command is used to verify that the VRF has been received by BGP.
Step 4	**show ip route vrf** *vrf-name* **Example:** `Device# show ip route vrf RED`	Displays the IP routing table associated with a VRF instance. The show ip route vrf command is used to verify that the VRF is in the routing table.

Configuration Examples for BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Example: Configuring eBGP and iBGP Multipath Load Sharing

This following configuration example configures a router in address-family mode to select six BGP routes (eBGP or iBGP) as multipaths:

```
Device(config)# router bgp 40000
 Device(config-router)# address-family ipv4 vrf RED
 Device(config-router-af)# maximum-paths eibgp 6
 Device(config-router-af)# end
```

Example: Verifying eBGP and iBGP Multipath Load Sharing

To verify that iBGP and eBGP routes have been configured for load sharing, use the **show ip bgp vpnv4** EXEC command or the **show ip route vrf** EXEC command.

In the following example, the **show ip bgp vpnv4** command is entered to display multipaths installed in the VPNv4 RIB:

```
Device# show ip bgp vpnv4 all 10.22.22.0

BGP routing table entry for 10:1:22.22.22.0/24, version 19
Paths:(5 available, best #5)
Multipath:eiBGP
  Advertised to non peer-group peers:
  10.0.0.2 10.0.0.3 10.0.0.4 10.0.0.5
  22
    10.0.0.2 (metric 20) from 10.0.0.4 (10.0.0.4)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:0x0:0:0 RT:100:1 0x0:0:0
      Originator:10.0.0.2, Cluster list:10.0.0.4
  22
    10.0.0.2 (metric 20) from 10.0.0.5 (10.0.0.5)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:0x0:0:0 RT:100:1 0x0:0:0
      Originator:10.0.0.2, Cluster list:10.0.0.5
  22
    10.0.0.2 (metric 20) from 10.0.0.2 (10.0.0.2)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:RT:100:1 0x0:0:0
  22
    10.0.0.2 (metric 20) from 10.0.0.3 (10.0.0.3)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:0x0:0:0 RT:100:1 0x0:0:0
      Originator:10.0.0.2, Cluster list:10.0.0.3
  22
    10.1.1.12 from 10.1.1.12 (10.22.22.12)
      Origin IGP, metric 0, localpref 100, valid, external, multipath, best
      Extended Community:RT:100:1
```

In the following example, the **show ip route vrf** command is entered to display multipath routes in the VRF table:

```
Device# show ip route vrf PATH 10.22.22.0

Routing entry for 10.22.22.0/24
  Known via "bgp 1", distance 20, metric 0
  Tag 22, type external
  Last update from 10.1.1.12 01:59:31 ago
  Routing Descriptor Blocks:
  * 10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.4, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.5, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.2, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.3, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.1.1.12, from 10.1.1.12, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
```

Additional References

Related Documents

Related Topic	Document Title
BGP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
Comprehensive BGP link bandwidth configuration examples and tasks	“BGP Link Bandwidth” module in the *IP Routing: BGP Configuration Guide*
CEF configuration tasks	“CEF Overview” module in the *IP Switching Cisco Express Forwarding Configuration Guide*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1771	*A Border Gateway Protocol 4 (BGP4)*
RFC 2547	*BGP/MPLS VPNs*
RFC 2858	*Multiprotocol Extensions for BGP-4*

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

CHAPTER 11

ORF Support for BGP Labeled Unicast

Table 7: Feature History

Feature Name	Release Information	Feature Description
Outbound Route Filtering (ORF) Support for BGP Labeled Unicast	Cisco IOS XE Cupertino 17.8.1	This feature uses BGP ORF send and receive capabilities to minimize the number of BGP updates that are sent between BGP peers. It can also filter out unwanted routing updates at the source to reduce the amount of system resources that are required for generating and processing routing updates.

This feature module describes how to configure BGP Prefix-Based Outbound Route Filtering (ORF) for BGP labeled unicast.

Overview of ORF Support for BGP Labeled Unicast

In earlier releases, BGP does not negotiate the labeled IPv4 unicast address family separately from the unlabeled IPv4 unicast address family. This results in outbound route filtering (ORF) negotiation failure when peering with other operating systems in the network.

Starting from Cisco IOS XE Cupertino 17.8.1, BGP uses the **label-unicast unique** mode. Unlabeled and labeled unicast BGP-prefixed sessions are treated separately. This improves the interoperability with other operating systems.

The BGP prefix-based ORF is enabled through the advertisement of ORF capabilities to peer routers. The advertisement of the ORF capability indicates that a BGP peer accepts a prefix list from a neighbor and applies the prefix list to locally configured ORFs (if any exist).

This feature uses BGP ORF send and receive capabilities to minimize the number of BGP updates that are sent between BGP peers. It can also filter out unwanted routing updates at the source to reduce the amount of system resources that are required for generating and processing routing updates.

Configure ORF Support for BGP Labeled Unicast

Before you begin

Enable this feature using the following command:

```
Router(config)# license boot level metroaggrservices
```

Procedure

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 **router bgp** *as-number*

Example:

```
Router(config)# router bgp 100
```

Enters router configuration mode.

- *as-number*—Number of an autonomous system that identifies the router to other BGP routers and tags the routing information that is passed along. The valid values range 1–65535.

Step 4 **neighbor** *ip-address* **update out labeled-unicast unique**

Example:

```
Router(config-router)# neighbor 3.3.3.3 update out labeled-unicast unique
```

Separates outbound processing for the two SAFIs (Subsequent Address Family Identifier). It creates two neighbor topologies to use ORF. Having the two neighbor topologies means you can negotiate the ORF capability separately per address-family.

Step 5 **neighbor** *ip-address* **update in labeled-unicast unique**

Example:

```
Router(config-router)# neighbor 3.3.3.3 update in labeled-unicast unique
```

Allows the router to accept one labeled path and one unlabeled path.

Step 6 **address family ipv4 unicast**

Example:

```
Router(config-router)# address family ipv4 unicast
```

Specifies the IPv4 unicast address family and enters address family configuration mode.

Step 7 **bgp mpls-local-label**

Example:

```
Router(config-router-af)# bgp mpls-locallabel
```

Enables BGP local label installation for unadvertised /32 prefixes.

Step 8 **neighbor** *ip-address* **send-label**

Example:

```
Router(config-router-af)# neighbor 3.3.3.3 send-label
```

Associates a BGP label to the prefix.

Step 9 **neighbor** *ip-address* **capability orf prefix-list [send|receive|both]**

Example:

```
Router(config-router-af)# neighbor 3.3.3.3 capability orf prefix-list both
```

Note The options **send**, **receive**, and **both** are supported only for individual neighbors under the **address-family-ipv4**. These options are not supported if a peer group is configured.

Enables the ORF capability on the local router, and enables ORF capability advertisement to the BGP peer specified with the *ip-address* argument.

- The **send** keyword configures a router to advertise ORF send capabilities.
- The **receive** keyword configures a router to advertise ORF receive capabilities.
- The **both** keyword configures a router to advertise send and receive capabilities.
- The remote peer must be configured to either send or receive ORF capabilities before outbound route filtering is enabled.
- The example configures the router to advertise send and receive capabilities to the 3.3.3.3 neighbor.

Configuration Example of ORF Support for BGP Labeled Unicast

The following is a sample configuration of the ORF Support for BGP Labeled Unicast feature.

```
router bgp 100
  neighbor 3.3.3.3 remote-as 100
  neighbor 3.3.3.3 update out labeled-unicast unique
  neighbor 3.3.3.3 update in labeled-unicast unique
  address-family ipv4 unicast
    bgp mpls-local-label
    neighbor 3.3.3.3 activate
    neighbor 3.3.3.3 send-label
    neighbor 3.3.3.3 capability orf prefix-list send|receive|both
    neighbor 3.3.3.3 prefix-list orf-prefix-list
ip prefix-list orf-prefix-list seq 10 permit 0.0.0.0/0
```

Verify the Configuration of ORF Support for BGP Labeled Unicast

To verify the configuration of ORF Support for BGP Labeled Unicast, use the following **show** command:

```
Router#show ip bgp ipv4 unicast neighbors 3.3.3.3
----Trimmed output ----
 Inbound updates with label-unicast (SAFI 4) considered unique respect unicast (SAFI 1):
 Outbound updates with label-unicast (SAFI 4) considered unique respect unicast (SAFI 1):

 For address family: IPv4 Label-Unicast
  Session: 3.3.3.3
  BGP table version 1, neighbor version 1/0
  Output queue size : 0
  Index 6, Advertise bit 1
  6 update-group member
  Inbound soft reconfiguration allowed
  NEXT_HOP is always this router for all paths
  Community attribute sent to this neighbor
  Extended-community attribute sent to this neighbor
  AF-dependant capabilities:
    Outbound Route Filter (ORF) type (128) Prefix-list:
      Send-mode: advertised, received
      Receive-mode: advertised, received
  Incoming update prefix filter list is PL_ORF
  Sending Prefix & Label
  Slow-peer detection is disabled
  Slow-peer split-update-group dynamic is disabled
                               Sent       Rcvd
  Prefix activity:             ----       ----
    Prefixes Current:             0          0
    Prefixes Total:               0          0
    Implicit Withdraw:            0          0
    Explicit Withdraw:            0          0
    Used as bestpath:           n/a          0
    Used as multipath:          n/a          0
    Used as secondary:          n/a          0
```

Use the following command on BGP peer which receives the prefix list from its neighbor.

```
Router#show ip bgp ipv4 unicast neighbors 10.0.0.1 received prefix-filter
Address family: IPv4 Unicast
ip prefix-list 10.0.0.1: 1 entries
seq 5 permit 23.23.23.23/32
```