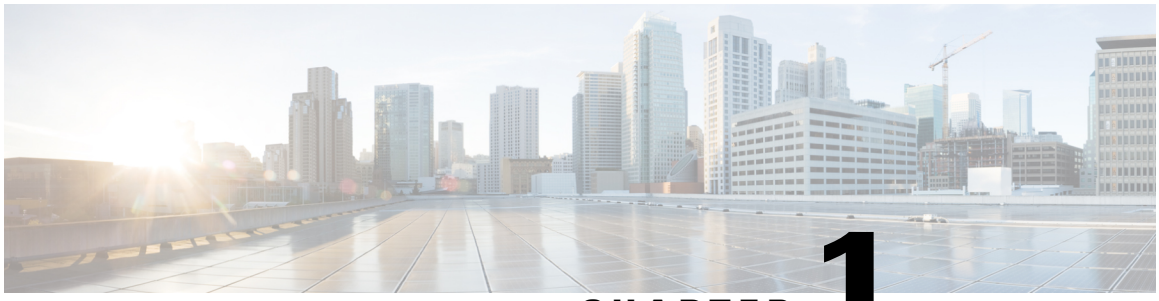
IP Routing: BGP Configuration Guide, Cisco IOS Release 15E

First Published: August 02, 2013

Last Modified: August 02, 2013

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

CONTENTS



CHAPTER 1

BGP Graceful Restart per Neighbor

The BGP graceful restart feature is already available on a global basis. The BGP Graceful Restart per Neighbor feature allows BGP graceful restart to be enabled or disable for an individual neighbor, providing greater network flexibility and service.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Graceful Restart per Neighbor

BGP Graceful Restart per Neighbor

The ability to enable or disable BGP graceful restart for every individual BGP neighbor was introduced. Three new methods of configuring BGP graceful restart for BGP peers, in addition to the existing global BGP graceful restart configuration, are now available. Graceful restart can be enabled or disabled for a BGP peer or a BGP peer group using the **neighbor ha-mode graceful-restart** command, or a BGP peer can inherit a graceful restart configuration from a BGP peer-session template using the **ha-mode graceful-restart** command.

Although BGP graceful restart is disabled by default, the existing global command enables graceful restart for all BGP neighbors regardless of their capabilities. The ability to enable or disable BGP graceful restart for individual BGP neighbors provides a greater level of control for a network administrator.

When the BGP graceful restart capability is configured for an individual neighbor, each method of configuring graceful restart has the same priority, and the last configuration instance is applied to the neighbor. For example, if global graceful restart is enabled for all BGP neighbors but an individual neighbor is subsequently configured as a member of a peer group for which the graceful restart is disabled, graceful restart is disabled for that neighbor.

The configuration of the restart and stale-path timers is available only with the global **bgp graceful-restart** command, but the default values are set when the **neighbor ha-mode graceful-restart** or **ha-mode graceful-restart** commands are configured. The default values are optimal for most network deployments, and these values should be adjusted only by an experienced network operator.

BGP Peer Session Templates

Peer session templates are used to group and apply the configuration of general BGP session commands to groups of neighbors that share session configuration elements. General session commands that are common for neighbors that are configured in different address families can be configured within the same peer session template. Peer session templates are created and configured in peer session configuration mode. Only general session commands can be configured in a peer session template.

General session commands can be configured once in a peer session template and then applied to many neighbors through the direct application of a peer session template or through indirect inheritance from a peer session template. The configuration of peer session templates simplifies the configuration of general session commands that are commonly applied to all neighbors within an autonomous system.

Peer session templates support direct and indirect inheritance. A BGP neighbor can be configured with only one peer session template at a time, and that peer session template can contain only one indirectly inherited peer session template. A BGP neighbor can directly inherit only one session template and can indirectly inherit up to seven additional peer session templates.

Peer session templates support inheritance. A directly applied peer session template can directly or indirectly inherit configurations from up to seven peer session templates. So, a total of eight peer session templates can be applied to a neighbor or neighbor group.

Peer session templates support only general session commands. BGP policy configuration commands that are configured only for a specific address family or NLRI configuration mode are configured with peer policy templates.

To use a BGP peer session template to enable or disable BGP graceful restart, see the “Enabling and Disabling BGP Graceful Restart Using BGP Peer Session Templates” section.

How to Configure BGP Graceful Restart per Neighbor

Enabling BGP Graceful Restart for an Individual BGP Neighbor

Perform this task on Router B in the figure above to enable BGP graceful restart on the internal BGP peer at Router C in the figure above. Under the IPv4 address family, the neighbor at Router C is identified, and BGP graceful restart is enabled for the neighbor at Router C with the IP address 172.21.1.2. To verify that BGP graceful restart is enabled, the optional **show ip bgp neighbors** command is used.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
5. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **activate**
7. **neighbor** *ip-address* **ha-mode graceful-restart** [**disable**]
8. **end**
9. **show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**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 router is placed in address family configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified. • 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.

	Command or Action	Purpose
Step 5	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 172.21.1.2 remote-as 45000`	Configures peering with a BGP neighbor in the specified autonomous system. • In this example, the BGP peer at 172.21.1.2 is an internal BGP peer because it has the same autonomous system number as the router where the BGP configuration is being entered (see Step 3).
Step 6	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 172.21.1.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 address family with the local router. • In this example, the internal BGP peer at 172.21.1.2 is activated.
Step 7	**neighbor** *ip-address* **ha-mode graceful-restart** [**disable**] **Example:** `Device(config-router-af)# neighbor 172.21.1.2 ha-mode graceful-restart`	Enables the BGP graceful restart capability for a BGP neighbor. • Use the **disable** keyword to disable BGP graceful restart capability. • If you enter this command after the BGP session has been established, you must restart the session in order for the capability to be exchanged with the BGP neighbor. • In this example, the BGP graceful restart capability is enabled for the neighbor at 172.21.1.2.
Step 8	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 9	**show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]] **Example:** `Device# show ip bgp neighbors 172.21.1.2`	(Optional) Displays information about TCP and BGP connections to neighbors. • "Graceful Restart Capability: advertised" will be displayed for each neighbor that has exchanged graceful restart capabilities with this router. • In this example, the output is filtered to display information about the BGP peer at 172.21.1.2.

Examples

The following example shows partial output from the **show ip bgp neighbors** command for the BGP peer at 172.21.1.2. Graceful restart is shown as enabled. Note the default values for the restart and stale-path timers. These timers can be set using only the global **bgp graceful-restart** command.

```
Device# show ip bgp neighbors 172.21.1.2
```

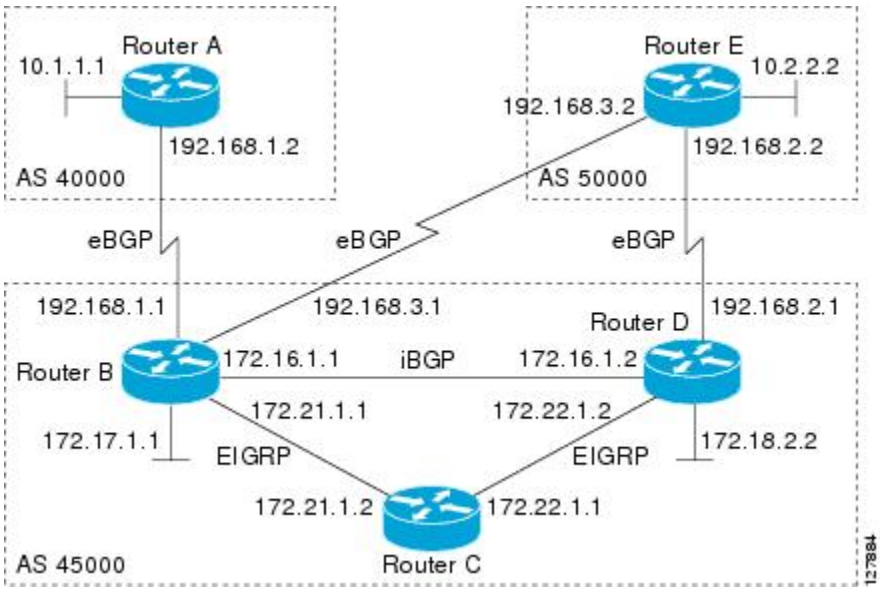
```
BGP neighbor is 172.21.1.2,  remote AS 45000, internal link
  BGP version 4, remote router ID 172.22.1.1
  BGP state = Established, up for 00:01:01
  Last read 00:00:02, last write 00:00:07, hold time is 180, keepalive intervals
  Neighbor sessions:
    1 active, is multisession capable
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Address family IPv4 Unicast: advertised and received
    Graceful Restart Capability: advertised
    Multisession Capability: advertised and received
!
  Address tracking is enabled, the RIB does have a route to 172.21.1.2
  Connections established 1; dropped 0
  Last reset never
  Transport(tcp) path-mtu-discovery is enabled
  Graceful-Restart is enabled, restart-time 120 seconds, stalepath-time 360 secs
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
```

Enabling and Disabling BGP Graceful Restart Using BGP Peer Session Templates

Perform this task to enable and disable BGP graceful restart for BGP neighbors using peer session templates. In this task, a BGP peer session template is created, and BGP graceful restart is enabled. A second peer session template is created, and this template is configured to disable BGP graceful restart.

In this example, the configuration is performed at Router B in the figure below, and two external BGP neighbors—Router A and Router E—are identified. The first BGP peer at Router A is configured to inherit the first peer session template, which enables BGP graceful restart, whereas the second BGP peer at Router E inherits the second template, which disables BGP graceful restart. Using the optional **show ip bgp neighbors** command, the status of the BGP graceful restart capability is verified for each BGP neighbor configured in this task.

Figure 1: Network Topology Showing BGP Neighbors



The restart and stale-path timers can be modified only using the global **bgp graceful-restart** command. The restart and stale-path timers are set to the default values when BGP graceful restart is enabled for BGP neighbors using peer session templates.

Note A BGP peer cannot inherit from a peer policy or session template and be configured as a peer group member at the same. BGP templates and BGP peer groups are mutually exclusive.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-session** *session-template-name*
5. **ha-mode graceful-restart** [**disable**]
6. **exit-peer-session**
7. **template peer-session** *session-template-name*
8. **ha-mode graceful-restart** [**disable**]
9. **exit-peer-session**
10. **bgp log-neighbor-changes**
11. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
12. **neighbor** *ip-address* **inherit peer-session** *session-template-number*
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **neighbor** *ip-address* **inherit peer-session** *session-template-number*
15. **end**
16. **show ip bgp template peer-session** [*session-template-number*]
17. **show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-session** *session-template-name* **Example:** `Device(config-router)# template peer-session S1`	Enters session-template configuration mode and creates a peer session template. • In this example, a peer session template named S1 is created.
Step 5	**ha-mode graceful-restart** [**disable**] **Example:** `Device(config-router-stmp)# ha-mode graceful-restart`	Enables the BGP graceful restart capability and BGP NSF awareness. • Use the **disable** keyword to disable BGP graceful restart capability. • If you enter this command after the BGP session has been established, you must restart the session in order for the capability to be exchanged with the BGP neighbor. • In this example, the BGP graceful restart capability is enabled for the peer session template named S1.
Step 6	**exit-peer-session** **Example:** `Device(config-router-stmp)# exit-peer-session`	Exits session-template configuration mode and returns to router configuration mode.
Step 7	**template peer-session** *session-template-name* **Example:** `Device(config-router)# template peer-session S2`	Enters session-template configuration mode and creates a peer session template. • In this example, a peer session template named S2 is created.
Step 8	**ha-mode graceful-restart** [**disable**] **Example:** `Device(config-router-stmp)# ha-mode graceful-restart disable`	Enables the BGP graceful restart capability and BGP NSF awareness. • Use the **disable** keyword to disable BGP graceful restart capability. • If you enter this command after the BGP session has been established, you must restart the session in order for the capability to be exchanged with the BGP neighbor. • In this example, the BGP graceful restart capability is disabled for the peer session template named S2.

	Command or Action	Purpose
Step 9	**exit-peer-session** **Example:** `Device(config-router-stmp)# exit-peer-session`	Exits session-template configuration mode and returns to router configuration mode.
Step 10	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	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 11	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Configures peering with a BGP neighbor in the specified autonomous system. • In this example, the BGP peer at 192.168.1.2 is an external BGP peer because it has a different autonomous system number from the router where the BGP configuration is being entered (see Step 3).
Step 12	**neighbor** *ip-address* **inherit peer-session** *session-template-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 inherit peer-session S1`	Inherits a peer session template. • In this example, the peer session template named S1 is inherited, and the neighbor inherits the enabling of BGP graceful restart.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.3.2 remote-as 50000`	Configures peering with a BGP neighbor in the specified autonomous system. • In this example, the BGP peer at 192.168.3.2 is an external BGP peer because it has a different autonomous system number from the router where the BGP configuration is being entered (see Step 3).
Step 14	**neighbor** *ip-address* **inherit peer-session** *session-template-number* **Example:** `Device(config-router)# neighbor 192.168.3.2 inherit peer-session S2`	Inherits a peer session-template. • In this example, the peer session template named S2 is inherited, and the neighbor inherits the disabling of BGP graceful restart.

	Command or Action	Purpose
Step 15	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 16	**show ip bgp template peer-session** [*session-template-number*] **Example:** `Device# show ip bgp template peer-session`	(Optional) Displays locally configured peer session templates. • The output can be filtered to display a single peer policy template by using the *session-template-name* argument. This command also supports all standard output modifiers.
Step 17	**show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about TCP and BGP connections to neighbors. • "Graceful Restart Capability: advertised" will be displayed for each neighbor that has exchanged graceful restart capabilities with this router. • In this example, the output is filtered to display information about the BGP peer at 192.168.1.2.

Examples

The following example shows partial output from the **show ip bgp neighbors** command for the BGP peer at 192.168.1.2 (Router A in the figure above). Graceful restart is shown as enabled. Note the default values for the restart and stale-path timers. These timers can be set only by using the **bgp graceful-restart** command.

```
Device# show ip bgp neighbors 192.168.1.2

BGP neighbor is 192.168.1.2,  remote AS 40000, external link
 Inherits from template S1 for session parameters
  BGP version 4, remote router ID 192.168.1.2
  BGP state = Established, up for 00:02:11
  Last read 00:00:23, last write 00:00:27, hold time is 180, keepalive intervals
  Neighbor sessions:
    1 active, is multisession capable
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Address family IPv4 Unicast: advertised and received
    Graceful Restart Capability: advertised
    Multisession Capability: advertised and received
!
Address tracking is enabled, the RIB does have a route to 192.168.1.2
  Connections established 1; dropped 0
  Last reset never
  Transport(tcp) path-mtu-discovery is enabled
  Graceful-Restart is enabled, restart-time 120 seconds, stalepath-time 360 secs
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
```

The following example shows partial output from the **show ip bgp neighbors** command for the BGP peer at 192.168.3.2 (Router E in the figure above). Graceful restart is shown as disabled.

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

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Inherits from template S2 for session parameters
  BGP version 4, remote router ID 192.168.3.2
  BGP state = Established, up for 00:01:41
  Last read 00:00:45, last write 00:00:45, hold time is 180, keepalive intervals
  Neighbor sessions:
    1 active, is multisession capable
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Address family IPv4 Unicast: advertised and received
!
Address tracking is enabled, the RIB does have a route to 192.168.3.2
  Connections established 1; dropped 0
  Last reset never
  Transport(tcp) path-mtu-discovery is enabled
  Graceful-Restart is disabled
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
```

Disabling BGP Graceful Restart for a BGP Peer Group

Perform this task to disable BGP graceful restart for a BGP peer group. In this task, a BGP peer group is created and graceful restart is disabled for the peer group. A BGP neighbor, Router D at 172.16.1.2 in the figure above, is then identified and added as a peer group member. It inherits the configuration associated with the peer group, which, in this example, disables BGP graceful restart.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
5. **neighbor** *peer-group-name* **peer-group**
6. **neighbor** *peer-group-name* **remote-as** *autonomous-system-number*
7. **neighbor** *peer-group-name* **ha-mode graceful-restart** [**disable**]
8. **neighbor** *ip-address* **peer-group** *peer-group-name*
9. **end**
10. **show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]

DETAILED STEPS

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

	Command or Action	Purpose
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 and creates a BGP routing process.
Step 4	**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 router is placed in address family configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified. • 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 5	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router-af)# neighbor PG1 peer-group`	Creates a BGP peer group. • In this example, the peer group named PG1 is created.
Step 6	**neighbor** *peer-group-name* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor PG1 remote-as 45000`	Configures peering with a BGP peer group in the specified autonomous system. • In this example, the BGP peer group named PG1 is added to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 7	**neighbor** *peer-group-name* **ha-mode graceful-restart** [**disable**] **Example:** `Device(config-router-af)# neighbor PG1 ha-mode graceful-restart disable`	Enables the BGP graceful restart capability for a BGP neighbor. • Use the **disable** keyword to disable BGP graceful restart capability. • If you enter this command after the BGP session has been established, you must restart the session for the capability to be exchanged with the BGP neighbor. • In this example, the BGP graceful restart capability is disabled for the BGP peer group named PG1.

	Command or Action	Purpose
Step 8	**neighbor** *ip-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 172.16.1.2 peer-group PG1`	Assigns the IP address of a BGP neighbor to a peer group. • In this example, the BGP neighbor peer at 172.16.1.2 is configured as a member of the peer group named PG1.
Step 9	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 10	**show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]] **Example:** `Device# show ip bgp neighbors 172.16.1.2`	(Optional) Displays information about TCP and BGP connections to neighbors. • In this example, the output is filtered to display information about the BGP peer at 172.16.1.2 and the "Graceful-Restart is disabled" line shows that the graceful restart capability is disabled for this neighbor.

Examples

The following example shows partial output from the **show ip bgp neighbors** command for the BGP peer at 172.16.1.2. Graceful restart is shown as disabled. Note the default values for the restart and stale-path timers. These timers can be set using only the global **bgp graceful-restart** command.

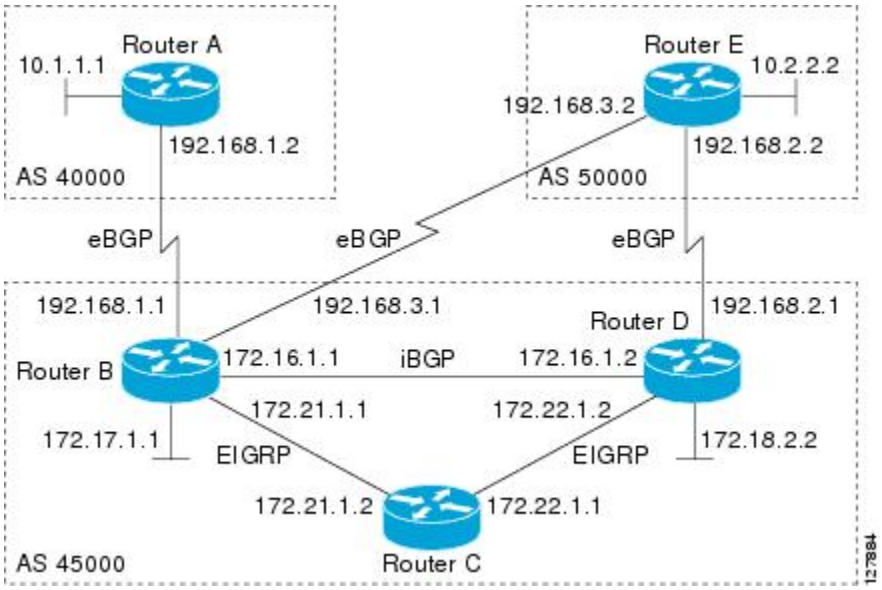
```
Device# show ip bgp neighbors 172.16.1.2

BGP neighbor is 172.16.1.2,  remote AS 45000, internal link
 Member of peer-group PG1 for session parameters
  BGP version 4, remote router ID 0.0.0.0
  BGP state = Idle
  Neighbor sessions:
    0 active, is multisession capable
!
Address tracking is enabled, the RIB does have a route to 172.16.1.2
  Connections established 0; dropped 0
  Last reset never
  Transport(tcp) path-mtu-discovery is enabled
  Graceful-Restart is disabled
```

Configuration Examples for BGP Graceful Restart per Neighbor

Examples: Enabling and Disabling BGP Graceful Restart per Neighbor

The ability to enable or disable the BGP graceful restart capability for an individual BGP neighbor, peer group, or peer session template was introduced. The following example is configured on Router B in the figure below and enables the BGP graceful restart capability for the BGP peer session template named S1 and disables the BGP graceful restart capability for the BGP peer session template named S2. The external BGP neighbor at Router A (192.168.1.2) inherits peer session template S1, and the BGP graceful restart capability is enabled for this neighbor. Another external BGP neighbor at Router E (192.168.3.2) is configured with the BGP graceful restart capability disabled after inheriting peer session template S2.

Figure 2: Network Topology Showing BGP Neighbors for BGP Graceful Restart



The BGP graceful restart capability is enabled for an individual internal BGP neighbor, Router C at 172.21.1.2, whereas the BGP graceful restart is disabled for the BGP neighbor at Router D, 172.16.1.2, because it is a member of the peer group PG1. The disabling of BGP graceful restart is configured for all members of the peer group, PG1. The restart and stale-path timers are modified, and the BGP sessions are reset.

```
router bgp 45000
 template peer-session S1
 remote-as 40000
 ha-mode graceful-restart
 exit-peer-session
 template peer-session S2
 remote-as 50000
 ha-mode graceful-restart disable
 exit-peer-session
 bgp log-neighbor-changes
 bgp graceful-restart restart-time 150
 bgp graceful-restart stalepath-time 400
 address-family ipv4 unicast
 neighbor PG1 peer-group
```

```
 neighbor PG1 remote-as 45000
 neighbor PG1 ha-mode graceful-restart disable
 neighbor 172.16.1.2 peer-group PG1
 neighbor 172.21.1.2 remote-as 45000
 neighbor 172.21.1.2 activate
 neighbor 172.21.1.2 ha-mode graceful-restart
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.1.2 inherit peer-session S1
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 inherit peer-session S2
 end
clear ip bgp *
```

To demonstrate how the last configuration instance of the BGP graceful restart capability is applied, the following example initially enables the BGP graceful restart capability globally for all BGP neighbors. A BGP peer group, PG2, is configured with the BGP graceful restart capability disabled. An individual external BGP neighbor, Router A at 192.168.1.2 in the figure above, is then configured to be a member of the peer group, PG2. The last graceful restart configuration instance is applied, and, in this case, the neighbor, 192.168.1.2, inherits the configuration instance from the peer group PG2, and the BGP graceful restart capability is disabled for this neighbor.

```
router bgp 45000
 bgp log-neighbor-changes
 bgp graceful-restart
 address-family ipv4 unicast
 neighbor PG2 peer-group
 neighbor PG2 remote-as 40000
 neighbor PG2 ha-mode graceful-restart disable
 neighbor 192.168.1.2 peer-group PG2
 end
clear ip bgp *
```

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 4724	*Graceful Restart Mechanism for BGP*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

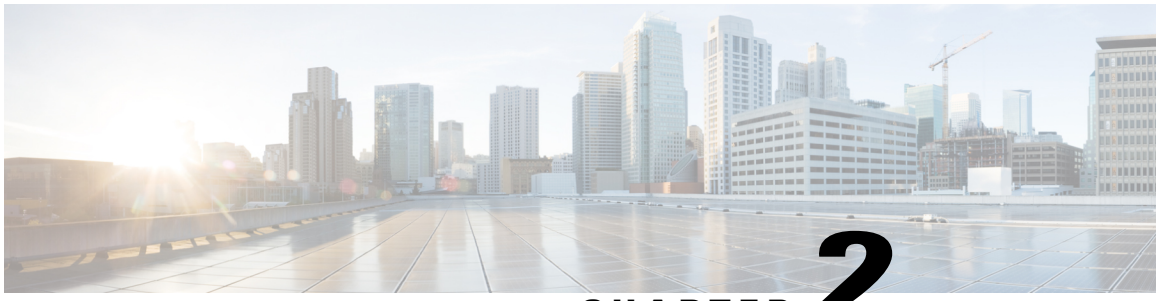
Feature Information for BGP Graceful Restart per Neighbor

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for BGP Graceful Restart per Neighbor

Feature Name	Releases	Feature Information
BGP Graceful Restart per Neighbor	15.1(1)SG	The BGP Graceful Restart per Neighbor feature enables or disables the BGP graceful restart capability for an individual BGP neighbor, including using peer session templates and BGP peer groups. The following commands were introduced by this feature: **ha-mode graceful-restart**, and**neighbor ha-mode graceful-restart**. The following command was modified by this feature: **show ip bgp neighbors**.



CHAPTER 2

BGP Support for Next-Hop Address Tracking

The BGP Support for Next-Hop Address Tracking feature is enabled by default when a supporting Cisco software image is installed. BGP next-hop address tracking is event driven. BGP prefixes are automatically tracked as peering sessions are established. Next-hop changes are rapidly reported to the BGP routing process as they are updated in the RIB. This optimization improves overall BGP convergence by reducing the response time to next-hop changes for routes installed in the RIB. When a bestpath calculation is run in between BGP scanner cycles, only next-hop changes are tracked and processed.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Support for Next-Hop Address Tracking

BGP Next-Hop Address Tracking

The BGP next-hop address tracking feature is enabled by default when a supporting Cisco software image is installed. BGP next-hop address tracking is event driven. BGP prefixes are automatically tracked as peering

sessions are established. Next-hop changes are rapidly reported to the BGP routing process as they are updated in the RIB. This optimization improves overall BGP convergence by reducing the response time to next-hop changes for routes installed in the RIB. When a best-path calculation is run in between BGP scanner cycles, only next-hop changes are tracked and processed.

Default BGP Scanner Behavior

BGP monitors the next hop of installed routes to verify next-hop reachability and to select, install, and validate the BGP best path. By default, the BGP scanner is used to poll the RIB for this information every 60 seconds. During the 60 second time period between scan cycles, Interior Gateway Protocol (IGP) instability or other network failures can cause black holes and routing loops to temporarily form.

BGP Next_Hop Attribute

The Next_Hop attribute identifies the next-hop IP address to be used as the BGP next hop to the destination. The router makes a recursive lookup to find the BGP next hop in the routing table. In external BGP (eBGP), the next hop is the IP address of the peer that sent the update. Internal BGP (iBGP) sets the next-hop address to the IP address of the peer that advertised the prefix for routes that originate internally. When any routes to iBGP that are learned from eBGP are advertised, the Next_Hop attribute is unchanged.

A BGP next-hop IP address must be reachable in order for the router to use a BGP route. Reachability information is usually provided by the IGP, and changes in the IGP can influence the forwarding of the next-hop address over a network backbone.

Selective BGP Next-Hop Route Filtering

BGP selective next-hop route filtering was implemented as part of the BGP Selective Address Tracking feature to support BGP next-hop address tracking. Selective next-hop route filtering uses a route map to selectively define routes to help resolve the BGP next hop.

The ability to use a route map with the **bgp nexthop** command allows the configuration of the length of a prefix that applies to the BGP Next_Hop attribute. The route map is used during the BGP bestpath calculation and is applied to the route in the routing table that covers the next-hop attribute for BGP prefixes. If the next-hop route fails the route map evaluation, the next-hop route is marked as unreachable. This command is per address family, so different route maps can be applied for next-hop routes in different address families.

Note Use route map on ASR series devices to set the next hop as BGP peer for the route and apply that route map in outbound direction towards the peer.

Note Only **match ip address** and **match source-protocol** commands are supported in the route map. No **set** commands or other **match** commands are supported.

BGP Support for Fast Peering Session Deactivation

BGP Hold Timer

By default, the BGP hold timer is set to run every 180 seconds in Cisco software. This timer value is set as the default to protect the BGP routing process from instability that can be caused by peering sessions with other routing protocols. BGP routers typically carry large routing tables, so frequent session resets are not desirable.

BGP Fast Peering Session Deactivation

BGP fast peering session deactivation improves BGP convergence and response time to adjacency changes with BGP neighbors. This feature is event driven and configured on a per-neighbor basis. When this feature is enabled, BGP will monitor the peering session with the specified neighbor. Adjacency changes are detected and terminated peering sessions are deactivated in between the default or configured BGP scanning interval.

Selective Address Tracking for BGP Fast Session Deactivation

In Cisco IOS XE Release 2.1 and later releases, the BGP Selective Address Tracking feature introduced the use of a route map with BGP fast session deactivation. The **route-map** keyword and *map-name* argument are used with the **neighbor fall-over** BGP neighbor session command to determine if a peering session with a BGP neighbor should be reset when a route to the BGP peer changes. The route map is evaluated against the new route, and if a deny statement is returned, the peer session is reset. The route map is not used for session establishment.

Note Only **match ip address** and **match source-protocol** commands are supported in the route map. No **set** commands or other **match** commands are supported.

How to Configure BGP Support for Next-Hop Address Tracking

Configuring BGP Next-Hop Address Tracking

The tasks in this section show how configure BGP next-hop address tracking. BGP next-hop address tracking significantly improves the response time of BGP to next-hop changes in the RIB. However, unstable Interior Gateway Protocol (IGP) peers can introduce instability to BGP neighbor sessions. We recommend that you aggressively dampen unstable IGP peering sessions to reduce the possible impact to BGP. For more details about configuring route dampening, see “Configuring BGP Route Dampening.”

Configuring BGP Selective Next-Hop Route Filtering

Perform this task to configure selective next-hop route filtering using a route map to filter potential next-hop routes. This task uses prefix lists and route maps to match IP addresses or source protocols and can be used

to avoid aggregate addresses and BGP prefixes being considered as next-hop routes. Only **match ip address** and **match source-protocol** commands are supported in the route map. No **set** commands or other **match** commands are supported.

For more examples of how to use the **bgp nexthop** command, see the "Examples: Configuring BGP Selective Next-Hop Route Filtering" section in this module.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
5. **bgp nexthop route-map** *map-name*
6. **exit**
7. **exit**
8. **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network / length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*]
9. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
10. **match ip address prefix-list** *prefix-list-name* [*prefix-list-name...*]
11. **exit**
12. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
13. **end**
14. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family ipv4** [**unicast**	**multicast**| **vrf** *vrf-name*]	Specifies the IPv4 address family and enters address family configuration mode.

	Command or Action	Purpose
	Example: `Device(config-router)# address-family ipv4 unicast`	• The **unicast** keyword specifies the IPv4 unicast address family. By default, the router is placed in address family 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 5	**bgp nexthop route-map** *map-name* **Example:** `Device(config-router-af)# bgp nexthop route-map CHECK-NEXTHOP`	Permits a route map to selectively define routes to help resolve the BGP next hop. • In this example the route map named CHECK-NEXTHOP is created.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 7	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 8	**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 FILTER25 seq 5 permit 0.0.0.0/0 le 25`	Creates a prefix list for BGP next-hop route filtering. • Selective next-hop route filtering supports prefix length matching or source protocol matching on a per address-family basis. • The example creates a prefix list named FILTER25 that permits routes only if the mask length is more than 25; this will avoid aggregate routes being considered as the next-hop route.
Step 9	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map CHECK-NEXTHOP deny 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named CHECK-NEXTHOP is created. If there is an IP address match in the following **match** command, the IP address will be denied.
Step 10	**match ip address prefix-list** *prefix-list-name* [*prefix-list-name...*]	Matches the IP addresses in the specified prefix list. • Use the *prefix-list-name* argument to specify the name of a prefix list. The ellipsis means that more than one prefix list can be specified.

	Command or Action	Purpose
	Example: `Device(config-route-map)# match ip address prefix-list FILTER25`	**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 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 12	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map CHECK-NEXTHOP permit 20`	Configures a route map and enters route map configuration mode. • In this example, all other IP addresses are permitted by route map CHECK-NEXTHOP.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	Displays the entries in the BGP routing table. • Enter this command to view the next-hop addresses for each route. **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*.

Example

The following example from the **show ip bgp** command shows the next-hop addresses for each route:

```
BGP table version is 7, 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
*  10.2.2.0/24      192.168.3.2              0             0 50000 i
*> 172.16.1.0/24    0.0.0.0                  0         32768 i
*> 172.17.1.0/24    0.0.0.0                  0         32768
```

Adjusting the Delay Interval for BGP Next-Hop Address Tracking

Perform this task to adjust the delay interval between routing table walks for BGP next-hop address tracking.

You can increase the performance of this feature by tuning the delay interval between full routing table walks to match the tuning parameters for the Interior Gateway protocol (IGP). The default delay interval is 5 seconds. This value is optimal for a fast-tuned IGP. In the case of an IGP that converges more slowly, you can change the delay interval to 20 seconds or more, depending on the IGP convergence time.

BGP next-hop address tracking significantly improves the response time of BGP to next-hop changes in the RIB. However, unstable Interior Gateway Protocol (IGP) peers can introduce instability to BGP neighbor sessions. We recommend that you aggressively dampen unstable IGP peering sessions to reduce the possible impact to BGP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [[**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]]
5. **bgp nexthop trigger delay** *delay-timer*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 64512`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**address-family ipv4** [[**mdt**	**multicast**	**tunnel**	**unicast** [**vrf** *vrf-name*]	**vrf** *vrf-name*]	**vpnv4** [**unicast**]]	Enter address family configuration mode to configure BGP peers to accept address family-specific configurations. • The example creates an IPv4 unicast address family session.

	Command or Action	Purpose
	Example: `Router(config-router)# address-family ipv4 unicast`	
Step 5	**bgp nexthop trigger delay** *delay-timer* **Example:** `Router(config-router-af)# bgp nexthop trigger delay 20`	Configures the delay interval between routing table walks for next-hop address tracking. • The time period determines how long BGP will wait before starting a full routing table walk after notification is received. • The value for the *delay-timer* argument is a number from 1 to 100 seconds. The default value is 5 seconds. • The example configures a delay interval of 20 seconds.
Step 6	**end** **Example:** `Router(config-router-af)# end`	Exits address-family configuration mode, and enters privileged EXEC mode.

Disabling BGP Next-Hop Address Tracking

Perform this task to disable BGP next-hop address tracking. BGP next-hop address tracking is enabled by default under the IPv4 and VPNv4 address families. Beginning with Cisco IOS Release 12.2(33)SB6, BGP next-hop address tracking is also enabled by default under the VPNv6 address family whenever the next hop is an IPv4 address mapped to an IPv6 next-hop address.

Disabling next hop address tracking may be useful if you the network has unstable IGP peers and route dampening is not resolving the stability issues. To reenable BGP next-hop address tracking, use the **bgp nexthop**command with the **trigger** and **enable** keywords.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [[**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**] | **vpnv6** [**unicast**]]
5. no **bgp nexthop trigger enable**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 64512`	Enters router configuration mod to create or configure a BGP routing process.
Step 4	**address-family ipv4** [[**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**] | **vpnv6** [**unicast**]] **Example:** `Router(config-router)# address-family ipv4 unicast`	Enter address family configuration mode to configure BGP peers to accept address family-specific configurations. • The example creates an IPv4 unicast address family session.
Step 5	no **bgp nexthop trigger enable** **Example:** `Router(config-router-af)# no bgp nexthop trigger enable`	Disables BGP next-hop address tracking. • Next-hop address tracking is enabled by default for IPv4 and VPNv4 address family sessions. • The example disables next-hop address tracking.
Step 6	**end** **Example:** `Router(config-router-af)# end`	Exits address-family configuration mode, and enters Privileged EXEC mode.

Configuring Fast Session Deactivation

The tasks in this section show how to configure BGP next-hop address tracking. BGP next-hop address tracking significantly improves the response time of BGP to next-hop changes in the RIB. However, unstable Interior Gateway Protocol (IGP) peers can introduce instability to BGP neighbor sessions. We recommend that you

aggressively dampen unstable IGP peering sessions to reduce the possible impact to BGP. For more details about route dampening, see the "Configuring Internal BGP Features" module.

Configuring Fast Session Deactivation for a BGP Neighbor

Perform this task to establish a peering session with a BGP neighbor and then configure the peering session for fast session deactivation to improve the network convergence time if the peering session is deactivated.

Enabling fast session deactivation for a BGP neighbor can significantly improve BGP convergence time. However, unstable IGP peers can still introduce instability to BGP neighbor sessions. We recommend that you aggressively dampen unstable IGP peering sessions to reduce the possible impact to BGP.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
5. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **fall-over**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 50000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Router(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure BGP peers to accept address family-specific configurations. • The example creates an IPv4 unicast address family session.

	Command or Action	Purpose
Step 5	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 10.0.0.1 remote-as 50000`	Establishes a peering session with a BGP neighbor.
Step 6	**neighbor** *ip-address* **fall-over** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 fall-over`	Configures the BGP peering to use fast session deactivation. • BGP will remove all routes learned through this peer if the session is deactivated.
Step 7	**end** **Example:** `Router(config-router-af)# end`	Exits configuration mode and returns to privileged EXEC mode.

Configuring Selective Address Tracking for Fast Session Deactivation

Perform this task to configure selective address tracking for fast session deactivation. The optional **route-map** keyword and *map-name* argument of the **neighbor fall-over** command are used to determine if a peering session with a BGP neighbor should be deactivated (reset) when a route to the BGP peer changes. The route map is evaluated against the new route, and if a deny statement is returned, the peer session is reset.

Note Only **match ip address** and **match source-protocol** commands are supported in the route map. No **set** commands or other **match** commands are supported.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number*
5. **neighbor** *ip-address* **fall-over** [**route-map** *map-name*]
6. **exit**
7. **ip prefix-list** *list-name* [**seq** *seq-value*]{**deny** *network / length* | **permit** *network / length*}[**ge** *ge-value*] [**le** *le-value*]
8. **route-map** *map-name* [**permit** | **deny**][*sequence-number*]
9. **match ip address prefix-list** *prefix-list-name* [*prefix-list-name...*]
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(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:** `Router(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 5	**neighbor** *ip-address* **fall-over** [**route-map** *map-name*]	Applies a route map when a route to the BGP changes. • In this example, the route map named CHECK-NBR is applied when the route to neighbor 192.168.1.2 changes.

	Command or Action	Purpose
	Example: `Router(config-router)# neighbor 192.168.1.2 fall-over route-map CHECK-NBR`	
Step 6	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 7	**ip prefix-list** *list-name* [**seq** *seq-value*]{**deny** *network / length* | **permit** *network / length*}[**ge** *ge-value*] [**le** *le-value*] **Example:** `Router(config)# ip prefix-list FILTER28 seq 5 permit 0.0.0.0/0 ge 28`	Creates a prefix list for BGP next-hop route filtering. • Selective next-hop route filtering supports prefix length matching or source protocol matching on a per-address-family basis. • The example creates a prefix list named FILTER28 that permits routes only if the mask length is greater than or equal to 28.
Step 8	**route-map** *map-name* [**permit** | **deny**][*sequence-number*] **Example:** `Router(config)# route-map CHECK-NBR permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named CHECK-NBR is created. If there is an IP address match in the following **match** command, the IP address will be permitted.
Step 9	**match ip address prefix-list** *prefix-list-name* [*prefix-list-name...*] **Example:** `Router(config-route-map)# match ip address prefix-list FILTER28`	Matches the IP addresses in the specified prefix list. • Use the *prefix-list-name* argument to specify the name of a prefix list. The ellipsis means that more than one prefix list can be specified. **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 10	**end** **Example:** `Router(config-route-map)# end`	Exits configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP Support for Next-Hop Address Tracking

Example: Enabling and Disabling BGP Next-Hop Address Tracking

In the following example, next-hop address tracking is disabled under the IPv4 address family session:

```
router bgp 50000
 address-family ipv4 unicast
 no bgp nexthop trigger enable
```

Example: Adjusting the Delay Interval for BGP Next-Hop Address Tracking

In the following example, the delay interval for next-hop tracking is configured to occur every 20 seconds under the IPv4 address family session:

```
router bgp 50000
 address-family ipv4 unicast
 bgp nexthop trigger delay 20
```

Examples: Configuring BGP Selective Next-Hop Route Filtering

The following example shows how to configure BGP selective next-hop route filtering to avoid using a BGP prefix as the next-hop route. If the most specific route that covers the next hop is a BGP route, then the BGP route will be marked as unreachable. The next hop must be an IGP or static route.

```
router bgp 45000
 address-family ipv4 unicast
 bgp nexthop route-map CHECK-BGP
 exit
 exit
route-map CHECK-BGP deny 10
 match source-protocol bgp 1
 exit
route-map CHECK-BGP permit 20
 end
```

The following example shows how to configure BGP selective next-hop route filtering to avoid using a BGP prefix as the next-hop route and to ensure that the prefix is more specific than /25.

```
router bgp 45000
 address-family ipv4 unicast
 bgp nexthop route-map CHECK-BGP25
 exit
 exit
ip prefix-list FILTER25 seq 5 permit 0.0.0.0/0 le 25
route-map CHECK-BGP25 deny 10
 match ip address prefix-list FILTER25
 exit
route-map CHECK-BGP25 deny 20
 match source-protocol bgp 1
 exit
route-map CHECK-BGP25 permit 30
 end
```

Example: Configuring Fast Session Deactivation for a BGP Neighbor

In the following example, the BGP routing process is configured on Router A and Router B to monitor and use fast peering session deactivation for the neighbor session between the two routers. Although fast peering session deactivation is not required at both routers in the neighbor session, it will help the BGP networks in both autonomous systems to converge faster if the neighbor session is deactivated.

Router A

```
router bgp 40000
 neighbor 192.168.1.1 remote-as 45000
 neighbor 192.168.1.1 fall-over
 end
```

Router B

```
router bgp 45000
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.1.2 fall-over
 end
```

Example: Configuring Selective Address Tracking for Fast Session Deactivation

The following example shows how to configure the BGP peering session to be reset if a route with a prefix of /28 or a more specific route to a peer destination is no longer available:

```
router bgp 45000
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.1.2 fall-over route-map CHECK-NBR
 exit
ip prefix-list FILTER28 seq 5 permit 0.0.0.0/0 ge 28
route-map CHECK-NBR permit 10
 match ip address prefix-list FILTER28
 end
```

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

Feature Information for BGP Support for Next-Hop Address Tracking

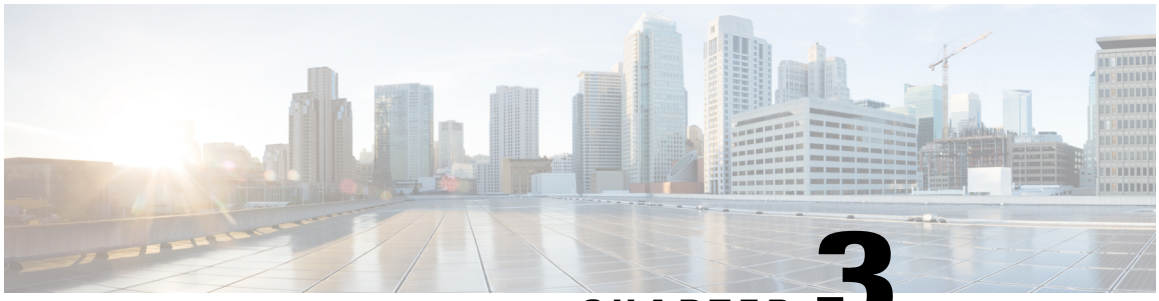
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for BGP Support for Next-Hop Address Tracking

Feature Name	Releases	Feature Information
BGP Support for Next-Hop Address Tracking	15.1(1)SG	The BGP Support for Next-Hop Address Tracking feature is enabled by default when a supporting Cisco IOS software image is installed. BGP next-hop address tracking is event driven. BGP prefixes are automatically tracked as peering sessions are established. Next-hop changes are rapidly reported to the BGP routing process as they are updated in the RIB. This optimization improves overall BGP convergence by reducing the response time to next-hop changes for routes installed in the RIB. When a bestpath calculation is run in between BGP scanner cycles, only next-hop changes are tracked and processed. The following command was introduced in this feature: **bgp nexthop**.
BGP Selective Address Tracking		The BGP Selective Address Tracking feature introduces the use of a route map for next-hop route filtering and fast session deactivation. Selective next-hop filtering uses a route map to selectively define routes to help resolve the BGP next hop, or a route map can be used to determine if a peering session with a BGP neighbor should be reset when a route to the BGP peer changes. The following commands were modified by this feature: **bgp nexthop**, **neighbor fall-over**.

Feature Name	Releases	Feature Information
BGP Support for Fast Peering Session Deactivation		The BGP Support for Fast Peering Session Deactivation feature introduced an event-driven notification system that allows a Border Gateway Protocol (BGP) process to monitor BGP peering sessions on a per-neighbor basis. This feature improves the response time of BGP to adjacency changes by allowing BGP to detect an adjacency change and deactivate the terminated session in between standard BGP scanning intervals. Enabling this feature improves overall BGP convergence. The following command was modified by this feature: **neighbor fall-over**.



CHAPTER 3

BGP Support for 4-byte ASN

The Cisco implementation of 4-byte autonomous system 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 autonomous system numbers in both the asplain format and the asdot format as described in RFC 5396.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About 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) will start in January 2009 to 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 3: 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 4: 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 5: 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

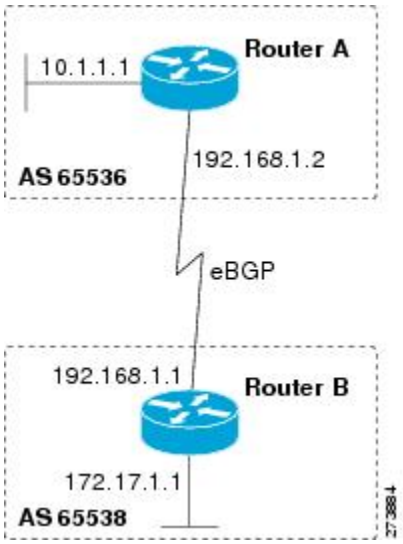
In Cisco IOS XE Release 2.4 and later releases, the Cisco implementation of 4-byte autonomous system numbers uses asplain—65538, for example—as the default regular expression match and output display format for autonomous system numbers, but you can configure 4-byte autonomous system numbers in both 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 autonomous system 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 autonomous system number formats, see the "BGP Autonomous System Number Formats" section.

In Cisco IOS XE Release 2.3, the Cisco implementation of 4-byte autonomous system 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 autonomous system numbers to 4-byte autonomous system numbers. To ensure a smooth transition, we recommend that all BGP speakers within an autonomous system that is identified using a 4-byte autonomous system number be upgraded to support 4-byte autonomous system numbers.

Note A new 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.

Figure 3: 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 BGP routing process and BGP peers when the BGP peers are located in an AS that uses 4-byte autonomous system 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 autonomous system numbers in this task are formatted in the default asplain (decimal value) format; for example, Router B is in autonomous system 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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. Repeat Step 4 to define other BGP neighbors, as required.
6. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
7. **neighbor** *ip-address* **activate**
8. Repeat Step 7 to activate other BGP neighbors, as required.
9. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
10. **end**
11. **show ip bgp** [*network*] [*network-mask*]
12. **show ip bgp summary**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte autonomous system number, 65538, is defined in asplain notation.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.2 remote-as 65536`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • In this example, the 4-byte autonomous system 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*]	Specifies the IPv4 address family and enters address family configuration mode.

	Command or Action	Purpose
	Example: `Router(config-router)# address-family ipv4 unicast`	• The **unicast** keyword specifies the IPv4 unicast address family. By default, the router 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 7	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af)# neighbor 192.168.1.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local router.
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:** `Router(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 10	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 11	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# 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:** `Router# 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 autonomous system 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 autonomous system 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 routers.

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 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 autonomous system numbers.

SUMMARY STEPS

1. **enable**
2. **show ip bgp summary**
3. **configure terminal**
4. **router bgp** *autonomous-system-number*
5. **bgp asnotation dot**
6. **end**
7. **clear ip bgp ***
8. **show ip bgp summary**
9. **show ip bgp regexp** *regexp*
10. **configure terminal**
11. **router bgp** *autonomous-system-number*
12. **no bgp asnotation dot**
13. **end**
14. **clear ip bgp ***

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp summary** **Example:** `Router# show ip bgp summary`	Displays the status of all BGP connections.
Step 3	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte autonomous system number, 65538, is defined in asplain notation.

	Command or Action	Purpose
Step 5	**bgp asnotation dot** **Example:** `Router(config-router)# bgp asnotation dot`	Changes the default output format of BGP 4-byte autonomous system numbers from asplain (decimal values) to dot notation. **Note** 4-byte autonomous system 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:** `Router(config-router)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 7	**clear ip bgp *** **Example:** `Router# clear ip bgp *`	Clears and resets all current BGP sessions. • In this example, a hard reset is performed to ensure that the 4-byte autonomous system 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:** `Router# show ip bgp summary`	Displays the status of all BGP connections.
Step 9	**show ip bgp regexp** *regexp* **Example:** `Router# show ip bgp regexp ^1\.0$`	Displays routes that match the autonomous system path regular expression. • In this example, a regular expression to match a 4-byte autonomous system path is configured using asdot format.
Step 10	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 11	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte autonomous system number, 65538, is defined in asplain notation.

	Command or Action	Purpose
Step 12	**no bgp asnotation dot** **Example:** `Router(config-router)# no bgp asnotation dot`	Resets the default output format of BGP 4-byte autonomous system numbers back to asplain (decimal values). **Note** 4-byte autonomous system 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:** `Router(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.
Step 14	**clear ip bgp *** **Example:** `Router# clear ip bgp *`	Clears and resets all current BGP sessions. • In this example, a hard reset is performed to ensure that the 4-byte autonomous system 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 autonomous system numbers. Note the asplain format of the 4-byte autonomous system 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 autonomous system numbers, 1.0 and 1.14 (these are the asdot conversions of the 65536 and 65550 autonomous system 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 autonomous system

paths is changed to asdot notation format. Although a 4-byte autonomous system number can be configured in a regular expression using either asplain format or asdot format, only 4-byte autonomous system 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 autonomous system 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 autonomous system 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

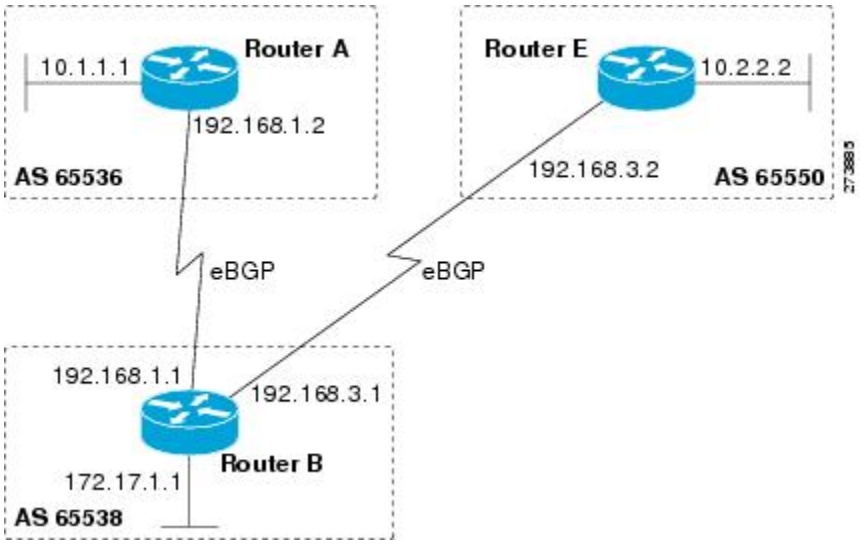
Examples: Configuring a BGP Routing Process and Peers Using 4-Byte Autonomous System Numbers

Asplain Format

The following example shows the configuration for Router A, Router B, and Router E in the figure below with a 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 4: 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
```

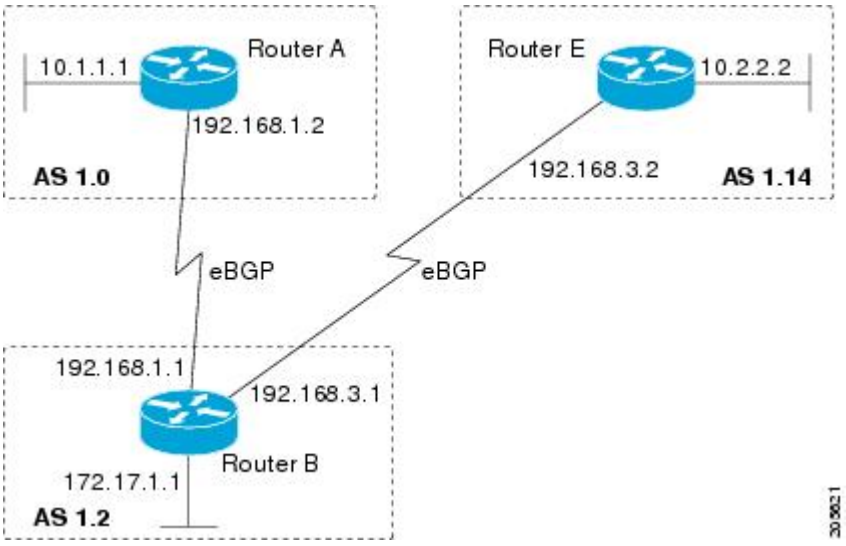
Router E

```
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 5: 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
```

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 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
```

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 4893	*BGP Support for Four-octet AS Number Space*

Standard/RFC	Title
RFC 5396	*Textual Representation of Autonomous System (AS) Numbers*
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

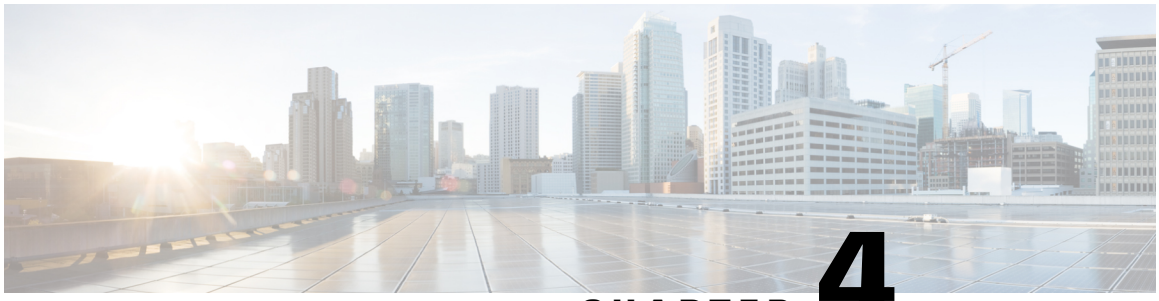
Feature Information for BGP Support for 4-byte ASN

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 6: Feature Information for BGP Support for 4-byte ASN

Feature Name	Releases	Feature Information
BGP Support for 4-byte ASN	15.1(1)SG 15.2(1)E	The BGP Support for 4-Byte ASN feature introduced support for 4-byte autonomous system numbers. The following commands were introduced or modified: **bgp asnotation dot**, **bgp confederation identifier**, **bgp confederation peers**, all **clear ip bgp**commands that configure an autonomous system number, **ip as-path access-list**, **ip extcommunity-list**, **match source-protocol**, **neighbor local-as**, **neighbor remote-as**, **redistribute (**IP**)**, **router bgp**, **route-target**, **set as-path**, **set extcommunity**, **set origin**, all **show ip bgp** commands that display an autonomous system number, and **show ip extcommunity-list**.



CHAPTER 4

BGP Support for Dual AS Configuration for Network AS Migrations

The BGP Support for Dual AS Configuration for Network AS Migrations feature extended the functionality of the BGP Local-AS feature by providing additional autonomous system path customization configuration options. The configuration of this feature is transparent to customer peering sessions, allowing the provider to merge two autonomous systems without interrupting customer peering arrangements. Customer peering sessions can later be updated during a maintenance window or during other scheduled downtime.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Support for Dual AS Configuration for Network AS Migrations

Autonomous System Migration for BGP Networks

Autonomous system migration can be necessary when a telecommunications or Internet service provider purchases another network. It is desirable for the provider to be able to integrate the second autonomous system without disrupting existing customer peering arrangements. The amount of configuration required in the customer networks can make this a cumbersome task that is difficult to complete without disrupting service.

Dual Autonomous System Support for BGP Network Autonomous System Migration

In Cisco IOS Release 12.0(29)S, 12.3(14)T, 12.2(33)SXH, and later releases, support was added for dual BGP autonomous system configuration to allow a secondary autonomous system to merge under a primary autonomous system, without disrupting customer peering sessions. The configuration of this feature is transparent to customer networks. Dual BGP autonomous system configuration allows a router to appear, to external peers, as a member of secondary autonomous system during the autonomous system migration. This feature allows the network operator to merge the autonomous systems and then later migrate customers to new configurations during normal service windows without disrupting existing peering arrangements.

The **neighbor local-as** command is used to customize the AS_PATH attribute by adding and removing autonomous system numbers for routes received from eBGP neighbors. This feature allows a router to appear to external peers as a member of another autonomous system for the purpose of autonomous system number migration. This feature simplifies this process of changing the autonomous system number in a BGP network by allowing the network operator to merge a secondary autonomous system into a primary autonomous system and then later update the customer configurations during normal service windows without disrupting existing peering arrangements.

BGP Autonomous System Migration Support for Confederations, Individual Peering Sessions, and Peer Groupings

This feature supports confederations, individual peering sessions, and configurations applied through peer groups and peer templates. If this feature is applied to group peers, the individual peers cannot be customized.

Ingress Filtering During BGP Autonomous System Migration

Autonomous system path customization increases the possibility that routing loops can be created if such customization is misconfigured. The larger the number of customer peerings, the greater the risk. You can minimize this possibility by applying policies on the ingress interfaces to block the autonomous system number that is in transition or routes that have no **local-as** configuration.

Caution BGP prepends the autonomous system number from each BGP network that a route traverses to maintain network reachability information and to prevent routing loops. This feature should be configured only for autonomous system migration and should be deconfigured after the transition has been completed. This procedure should be attempted only by an experienced network operator, as routing loops can be created with improper configuration.

BGP Network Migration to 4-Byte Autonomous System Numbers

The BGP Support for 4-Byte ASN feature introduced support for 4-byte autonomous system numbers. Because of increased demand for autonomous system numbers, in January 2009 the IANA started to allocate 4-byte autonomous system numbers in the range from 65536 to 4294967295.

The Cisco implementation of 4-byte autonomous system numbers 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.

Migrating your BGP network to 4-byte autonomous system numbers requires some planning. 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.

To ensure a smooth transition, we recommend that all BGP speakers within an autonomous system that is identified using a 4-byte autonomous system number be upgraded to support 4-byte autonomous system numbers.

For details about steps to perform to upgrade a BGP network to full 4-byte autonomous system support, see the Migration Guide for Explaining 4-Byte Autonomous System white paper.

How to Configure BGP Support for Dual AS Configuration for Network AS Migrations

Configuring Dual AS Peering for Network Migration

Perform this task to configure a BGP peer router to appear to external peers as a member of another autonomous system for the purpose of autonomous system number migration. When the BGP peer is configured with dual autonomous system numbers then the network operator can merge a secondary autonomous system into a primary autonomous system and update the customer configuration during a future service window without disrupting existing peering arrangements.

The **show ip bgp** and **show ip bgp neighbors** commands can be used to verify autonomous system number for entries in the routing table and the status of this feature.

Note

- The BGP Support for Dual AS Configuration for Network AS Migrations feature can be configured for only true eBGP peering sessions. This feature cannot be configured for two peers in different subautonomous systems of a confederation.
- The BGP Support for Dual AS Configuration for Network AS Migrations feature can be configured for individual peering sessions and configurations applied through peer groups and peer templates. If this command is applied to a peer group, the peers cannot be individually customized.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. **neighbor** *ip-address* **local-as** [*autonomous-system-number* [**no-prepend** [**replace-as** [**dual-as**]]]]
6. **neighbor** *ip-address* **remove-private-as**
7. **end**
8. **show ip bgp** [*network*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter-prefixes** *mask-length*]
9. **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 40000`	Enters router configuration mode, and creates a BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number*	Establishes a peering session with a BGP neighbor.

	Command or Action	Purpose
	Example: `Router(config-router)# neighbor 10.0.0.1 remote-as 45000`	
Step 5	**neighbor** *ip-address* **local-as** [*autonomous-system-number* [**no-prepend** [**replace-as** [**dual-as**]]]] **Example:** `Router(config-router)# neighbor 10.0.0.1 local-as 50000 no-prepend replace-as dual-as`	Customizes the AS_PATH attribute for routes received from an eBGP neighbor. • The **replace-as** keyword is used to prepend only the local autonomous system number (as configured with the *ip-address* argument) to the AS_PATH attribute. The autonomous system number from the local BGP routing process is not prepended. • The **dual-as** keyword is used to configure the eBGP neighbor to establish a peering session using the real autonomous-system number (from the local BGP routing process) or by using the autonomous system number configured with the *ip-address* argument (local-as). • The example configures the peering session with the 10.0.0.1 neighbor to accept the real autonomous system number and the local-as number.
Step 6	**neighbor** *ip-address* **remove-private-as** **Example:** `Router(config-router)# neighbor 10.0.0.1 remove-private-as`	(Optional) Removes private autonomous system numbers from outbound routing updates. • This command can be used with the **replace-as** functionality to remove the private autonomous system number and replace it with an external autonomous system number. • Private autonomous system numbers (64512 to 65535) are automatically removed from the AS_PATH attribute when this command is configured.
Step 7	**end** **Example:** `Router(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 8	**show ip bgp** [*network*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter-prefixes** *mask-length*] **Example:** `Router# show ip bgp`	Displays entries in the BGP routing table. • The output can be used to verify if the real autonomous system number or local-as number is configured.

	Command or Action	Purpose
Step 9	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*] **Example:** `Router# show ip bgp neighbors`	Displays information about TCP and BGP connections to neighbors. • The output will display **local AS**, **no-prepend**, **replace-as**, and **dual-as** with the corresponding autonomous system number when these options are configured.

Configuration Examples for Dual-AS Peering for Network Migration

Example: Dual AS Configuration

The following examples shows how this feature is used to merge two autonomous systems without interrupting peering arrangements with the customer network. The **neighbor local-as** command is configured to allow Router 1 to maintain peering sessions through autonomous system 40000 and autonomous system 45000. Router 2 is a customer router that runs a BGP routing process in autonomous system 50000 and is configured to peer with autonomous-system 45000.

Router 1 in Autonomous System 40000 (Provider Network)

```
interface Serial3/0
 ip address 10.3.3.11 255.255.255.0
!
router bgp 40000
 no synchronization
 bgp router-id 10.0.0.11
 neighbor 10.3.3.33 remote-as 50000
 neighbor 10.3.3.33 local-as 45000 no-prepend replace-as dual-as
```

Router 1 in Autonomous System 45000 (Provider Network)

```
interface Serial3/0
 ip address 10.3.3.11 255.255.255.0
!
router bgp 45000
 bgp router-id 10.0.0.11
 neighbor 10.3.3.33 remote-as 50000
```

Router 2 in Autonomous System 50000 (Customer Network)

```
interface Serial3/0
 ip address 10.3.3.33 255.255.255.0
!
router bgp 50000
```

```
 bgp router-id 10.0.0.3
 neighbor 10.3.3.11 remote-as 45000
```

After the transition is complete, the configuration on router 50000 can be updated to peer with autonomous system 40000 during a normal maintenance window or during other scheduled downtime:

```
 neighbor 10.3.3.11 remote-as 100
```

Example: Dual AS Confederation Configuration

The following example can be used in place of the Router 1 configuration in the "Example: Dual AS Configuration" example. The only difference between these configurations is that Router 1 is configured to be part of a confederation.

```
interface Serial3/0/0
 ip address 10.3.3.11 255.255.255.0
!
router bgp 65534
 no synchronization
 bgp confederation identifier 100
 bgp router-id 10.0.0.11
 neighbor 10.3.3.33 remote-as 50000
 neighbor 10.3.3.33 local-as 45000 no-prepend replace-as dual-as
```

Example: Replace an AS with Another AS in Routing Updates

The following example strips private autonomous system 64512 from outbound routing updates for the 10.3.3.33 neighbor and replaces it with autonomous system 50000:

```
router bgp 64512
 neighbor 10.3.3.33 local-as 50000 no-prepend replace-as
```

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

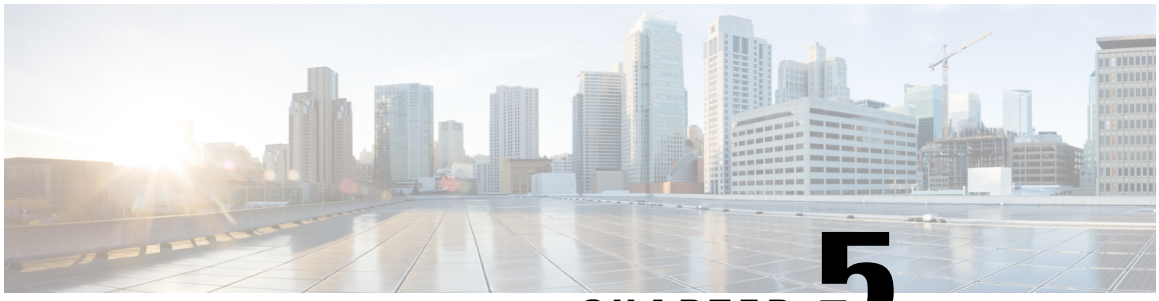
Feature Information for BGP Support for Dual AS Configuration for Network AS Migrations

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 7: Feature Information for BGP Support for Dual AS Configuration for Network AS Migrations

Feature Name	Releases	Feature Information
BGP Support for Dual AS Configuration for Network AS Migrations	15.1(1)SG	The BGP Support for Dual AS Configuration for Network AS Migrations feature extended the functionality of the BGP Local-AS feature by providing additional autonomous system path customization configuration options. The configuration of this feature is transparent to customer peering sessions, allowing the provider to merge two autonomous systems without interrupting customer peering arrangements. Customer peering sessions can later be updated during a maintenance window or during other scheduled downtime. The following command was modified by this feature: **neighbor local-as**.



CHAPTER 5

BGP Dynamic Neighbors

BGP dynamic neighbor support allows BGP peering to a group of remote neighbors that are defined by a range of IP addresses. Each range can be configured as a subnet IP address. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Dynamic Neighbors

BGP Dynamic Neighbors

Support for the BGP Dynamic Neighbors feature was introduced in Cisco IOS Release 12.2(33)SXH on the Cisco Catalyst 6500 series switches. BGP dynamic neighbor support allows BGP peering to a group of remote neighbors that are defined by a range of IP addresses. Each range can be configured as a subnet IP address. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups.

After a subnet range is configured for a BGP peer group and a TCP session is initiated by another router for an IP address in the subnet range, a new BGP neighbor is dynamically created as a member of that group. After the initial configuration of subnet ranges and activation of the peer group (referred to as a *listen range group*), dynamic BGP neighbor creation does not require any further CLI configuration on the initial router. Other routers can establish a BGP session with the initial router, but the initial router need not establish a BGP session to other routers if the IP address of the remote peer used for the BGP session is not within the configured range.

To support the BGP Dynamic Neighbors feature, the output for the **show ip bgp neighbors**, **show ip bgp peer-group**, and **show ip bgp summary** commands was updated to display information about dynamic neighbors.

A dynamic BGP neighbor will inherit any configuration for the peer group. In larger BGP networks, implementing BGP dynamic neighbors can reduce the amount and complexity of CLI configuration and save CPU and memory usage. Only IPv4 peering is supported.

How to Configure BGP Dynamic Neighbors

Implementing BGP Dynamic Neighbors Using Subnet Ranges

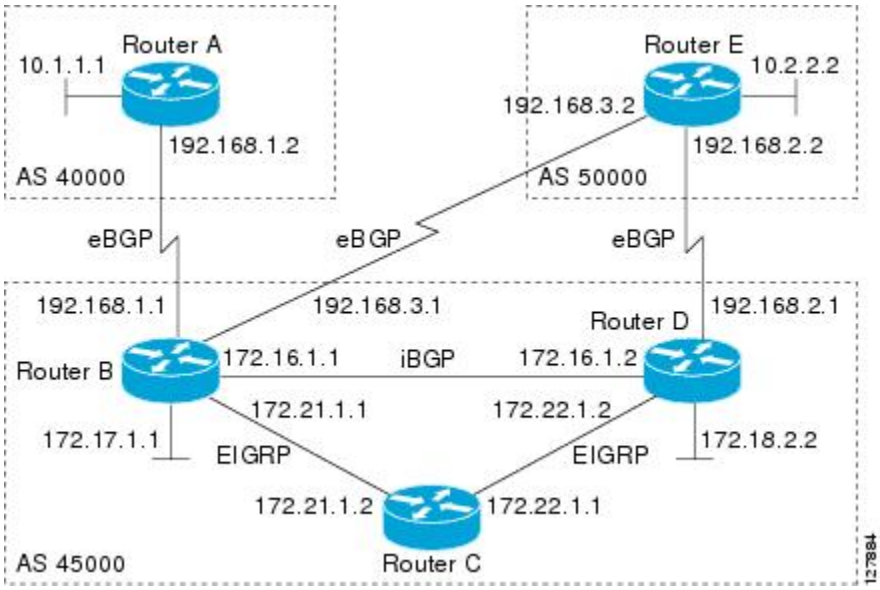
In Cisco IOS Release 12.2(33)SXH, support for BGP dynamic neighbors was introduced. Perform this task to implement the dynamic creation of BGP neighbors using subnet ranges.

In this task, a BGP peer group is created on Router B in the figure below, a global limit is set on the number of dynamic BGP neighbors, and a subnet range is associated with a peer group. Configuring the subnet range enables the dynamic BGP neighbor process. The peer group is added to the BGP neighbor table of the local router, and an alternate autonomous system number is also configured. The peer group is activated under the IPv4 address family.

The next step is to move to another router—Router E in the figure below—where a BGP session is started and the neighbor router, Router B, is configured as a remote BGP peer. The peering configuration opens a TCP session and triggers Router B to create a dynamic BGP neighbor because the IP address that starts the TCP session (192.168.3.2) is within the configured subnet range for dynamic BGP peers. The task moves

back to the first router, Router B, to run three **show** commands that have been modified to display dynamic BGP peer information.

Figure 6: BGP Dynamic Neighbor Topology



Before You Begin

This task requires Cisco IOS Release 12.2(33)SXH, or a later release, to be running.

Note This task supports only IPv4 BGP peering.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp log-neighbor-changes**
5. **neighbor** *peer-group-name* **peer-group**
6. **bgp listen** [**limit** *max-number*]
7. **bgp listen** [**limit** *max-number* | **range** *network* / *length* **peer-group** *peer-group-name*]
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **ebgp-multihop** [*ttl*]
9. **neighbor** *peer-group-name* **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]
10. **address-family ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]]
11. **neighbor** {*ip-address* | *peer-group-name*} **activate**
12. **end**
13. Move to another router that has an interface within the subnet range for the BGP peer group configured in this task.
14. **enable**
15. **configure terminal**
16. **router bgp** *autonomous-system-number*
17. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]
18. Return to the first router.
19. **show ip bgp summary**
20. **show ip bgp peer-group** [*peer-group-name*] [**summary**]
21. **show ip bgp neighbors** [*ip-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `DeviceB> enable`	Enables privileged EXEC mode. • Enter your password if prompted. • The configuration is entered on router B.
Step 2	**configure terminal** **Example:** `DeviceB# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `DeviceB(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**bgp log-neighbor-changes** **Example:** `DeviceB(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 5	**neighbor** *peer-group-name* **peer-group** **Example:** `DeviceB(config-router)# neighbor group192 peer-group`	Creates a BGP peer group. • In this example, a peer group named group192 is created. This group will be used as a listen range group.
Step 6	**bgp listen** [**limit** *max-number*] **Example:** `DeviceB(config-router)# bgp listen limit 200`	Sets a global limit of BGP dynamic subnet range neighbors. • Use the optional **limit** keyword and *max-number* argument to define the maximum number of BGP dynamic subnet range neighbors that can be created. **Note** Only the syntax applicable to this task is used in this example. For the complete syntax, see Step 7.
Step 7	**bgp listen** [**limit** *max-number* | **range** *network* / *length* **peer-group** *peer-group-name*] **Example:** `DeviceB(config-router)# bgp listen range 192.168.0.0/16 peer-group group192`	Associates a subnet range with a BGP peer group and activates the BGP dynamic neighbors feature. • Use the optional **limit** keyword and *max-number* argument to define the maximum number of BGP dynamic neighbors that can be created. • Use the optional **range** keyword and *network* / *length* argument to define a prefix range to be associated with the specified peer group. • In this example, the prefix range 192.168.0.0/16 is associated with the listen range group named group192.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `DeviceB(config-router)# neighbor group192 ebgp-multihop 255`	Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.

	Command or Action	Purpose
Step 9	**neighbor** *peer-group-name* **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `DeviceB(config-router)# neighbor group192 remote-as 40000 alternate-as 50000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • Use the optional **alternate-as** keyword and *autonomous-system-number* argument to identify up to five alternate autonomous system numbers for listen range neighbors. • In this example, the peer group named group192 is configured with two possible autonomous system numbers. **Note** The **alternate-as** keyword is used only with the listen range peer groups, not with individual BGP neighbors.
Step 10	**address-family ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]] **Example:** `DeviceB(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `DeviceB(config-router-af)# neighbor group192 activate`	Activates the neighbor or listen range peer group for the configured address family. • In this example, the neighbor 172.16.1.1 is activated for the IPv4 address family. **Note** Usually BGP peer groups cannot be activated using this command, but the listen range peer groups are a special case.
Step 12	**end** **Example:** `DeviceB(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 13	Move to another router that has an interface within the subnet range for the BGP peer group configured in this task.	—
Step 14	**enable** **Example:** `DeviceE> enable`	Enables privileged EXEC mode. • Enter your password if prompted. • The configuration is entered on Router E.
Step 15	**configure terminal** **Example:** `DeviceE# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 16	**router bgp** *autonomous-system-number* **Example:** `DeviceE(config)# router bgp 50000`	Enters router configuration mode for the specified routing process.
Step 17	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `DeviceE(config-router)# neighbor 192.168.3.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • In this example, the interface (192.168.3.2 in the figure above) at Router E is with the subnet range set for the BGP listen range group, group192. When TCP opens a session to peer to Router B, Router B creates this peer dynamically.
Step 18	Return to the first router.	—
Step 19	**show ip bgp summary** **Example:** `DeviceB# show ip bgp summary`	(Optional) Displays the BGP path, prefix, and attribute information for all connections to BGP neighbors. • In this step, the configuration has returned to Router B.
Step 20	**show ip bgp peer-group** [*peer-group-name*] [**summary**] **Example:** `DeviceB# show ip bgp peer-group group192`	(Optional) Displays information about BGP peer groups.
Step 21	**show ip bgp neighbors** [*ip-address*] **Example:** `DeviceB# show ip bgp neighbors 192.168.3.2`	(Optional) Displays information about BGP and TCP connections to neighbors. • In this example, information is displayed about the dynamically created neighbor at 192.168.3.2. The IP address of this BGP neighbor can be found in the output of either the **show ip bgp summary** or the **show ip bgp peer-group** command. **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 examples were taken from Router B in the figure above after the appropriate configuration steps in this task were completed on both Router B and Router E.

The following output from the **show ip bgp summary** command shows that the BGP neighbor 192.168.3.2 was dynamically created and is a member of the listen range group, group192. The output also shows that the IP prefix range of 192.168.0.0/16 is defined for the listen range named group192.

```
Router# show ip bgp summary
BGP router identifier 192.168.3.1, local AS number 45000
BGP table version is 1, main routing table version 1
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
*192.168.3.2    4 50000       2       2        0    0    0 00:00:37        0
* Dynamically created based on a listen range command
Dynamically created neighbors: 1/(200 max), Subnet ranges: 1
BGP peergroup group192 listen range group members:
  192.168.0.0/16
```

The following output from the **show ip bgp peer-group** command shows information about the listen range group, group192 that was configured in this task:

```
Router# show ip bgp peer-group group192
BGP peer-group is group192,  remote AS 40000
  BGP peergroup group192 listen range group members:
  192.168.0.0/16
  BGP version 4
  Default minimum time between advertisement runs is 30 seconds
 For address family: IPv4 Unicast
  BGP neighbor is group192, peer-group external, members:
  *192.168.3.2
  Index 0, Offset 0, Mask 0x0
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

The following sample output from the **show ip bgp neighbors** command shows that the neighbor 192.168.3.2 is a member of the peer group, group192, and belongs to the subnet range group 192.168.0.0/16, which shows that this peer was dynamically created:

```
Router# show ip bgp neighbors 192.168.3.2
BGP neighbor is *192.168.3.2,  remote AS 50000, external link
 Member of peer-group group192 for session parameters
 Belongs to the subnet range group: 192.168.0.0/16
  BGP version 4, remote router ID 192.168.3.2
  BGP state = Established, up for 00:06:35
  Last read 00:00:33, last write 00:00:25, hold time is 180, keepalive intervals
  Neighbor capabilities:
    Route refresh: advertised and received(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:                0          0
    Keepalives:             7          7
    Route Refresh:          0          0
    Total:                  8          8
  Default minimum time between advertisement runs is 30 seconds
 For address family: IPv4 Unicast
  BGP table version 1, neighbor version 1/0
  Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
  group192 peer-group member
.
.
.
```

Configuration Examples for BGP Dynamic Neighbors

Example: Implementing BGP Dynamic Neighbors Using Subnet Ranges

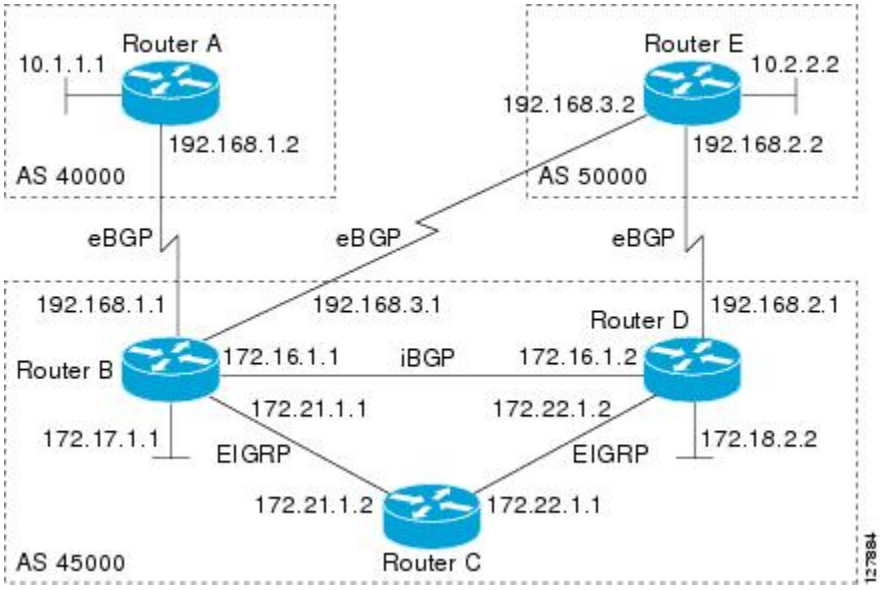
In the following example, two BGP peer groups are created on Router B in the figure below, a global limit is set on the number of dynamic BGP neighbors, and a subnet range is associated with a peer group. Configuring the subnet range enables the dynamic BGP neighbor process. The peer groups are added to the BGP neighbor table of the local router, and an alternate autonomous system number is also configured for one of the peer groups, group192. The subnet range peer groups and a standard BGP peer are then activated under the IPv4 address family.

The configuration moves to another router—Router A in the figure below—where a BGP session is started and the neighbor router, Router B, is configured as a remote BGP peer. The peering configuration opens a TCP session and triggers Router B to create a dynamic BGP neighbor because the IP address that starts the TCP session (192.168.1.2) is within the configured subnet range for dynamic BGP peers.

A third router—Router E in the figure below—also starts a BGP peering session with Router B. Router E is in the autonomous system 50000, which is the configured alternate autonomous system. Router B responds to the resulting TCP session by creating another dynamic BGP peer.

This example concludes with the output of the **show ip bgp summary** command entered on Router B.

Figure 7: BGP Dynamic Neighbor Topology



Router B

```
enable
configure terminal
router bgp 45000
 bgp log-neighbor-changes
 bgp listen limit 200
 bgp listen range 172.21.0.0/16 peer-group group172
```

```
 bgp listen range 192.168.0.0/16 peer-group group192
 neighbor group172 peer-group
 neighbor group172 remote-as 45000
 neighbor group192 peer-group
 neighbor group192 remote-as 40000 alternate-as 50000
 neighbor 172.16.1.2 remote-as 45000
 address-family ipv4 unicast
 neighbor group172 activate
 neighbor group192 activate
 neighbor 172.16.1.2 activate
 end
```

Router A

```
enable
configure terminal
router bgp 40000
 neighbor 192.168.1.1 remote-as 45000
 exit
```

Router E

```
enable
configure terminal
router bgp 50000
 neighbor 192.168.3.1 remote-as 45000
 exit
```

After both Router A and Router E are configured, the **show ip bgp summary** command is run on Router B. The output displays the regular BGP neighbor, 172.16.1.2, and the two BGP neighbors that were created dynamically when Router A and Router E initiated TCP sessions for BGP peering to Router B. The output also shows information about the configured listen range subnet groups.

```
BGP router identifier 192.168.3.1, local AS number 45000
BGP table version is 1, main routing table version 1
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
172.16.1.2      4 45000      15      15        1    0    0 00:12:20        0
*192.168.1.2    4 40000       3       3        1    0    0 00:00:37        0
*192.168.3.2    4 50000       6       6        1    0    0 00:04:36        0
* Dynamically created based on a listen range command
Dynamically created neighbors: 2/(200 max), Subnet ranges: 2
BGP peergroup group172 listen range group members:
  172.21.0.0/16
BGP peergroup group192 listen range group members:
  192.168.0.0/16
```

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 2918	*Route Refresh Capability 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

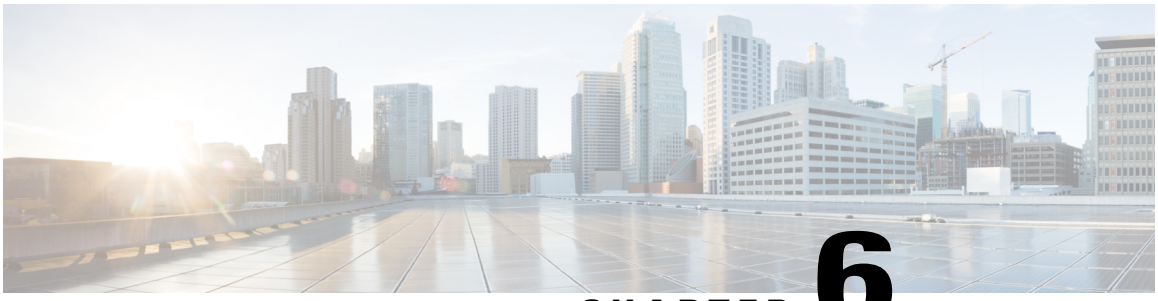
Feature Information for BGP Dynamic Neighbors

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 8: Feature Information for BGP Dynamic Neighbors

Feature Name	Releases	Feature Information
BGP Dynamic Neighbors	15.1(1)SG	BGP dynamic neighbor support allows BGP peering to a group of remote neighbors that are defined by a range of IP addresses. Each range can be configured as a subnet IP address. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups. After a subnet range is configured for a BGP peer group and a TCP session is initiated for an IP address in the subnet range, a new BGP neighbor is dynamically created as a member of that group. The new BGP neighbor will inherit any configuration for the peer group. The following commands were introduced or modified by this feature: **bgp listen**, **debug ip bgp range**, **neighbor remote-as**, **show ip bgp neighbors**, **show ip bgp peer-group**, and **show ip bgp summary**.



CHAPTER 6

BGP Support for the L2VPN Address Family

BGP support for the Layer 2 Virtual Private Network (L2VPN) address family introduces a BGP-based autodiscovery mechanism to distribute L2VPN endpoint provisioning information. BGP uses a separate L2VPN Routing Information Base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 virtual forwarding instance (VFI) is configured. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to set up a pseudowire mesh to support L2VPN-based services.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for BGP Support for the L2VPN Address Family

The BGP Support for the L2VPN Address Family feature assumes prior knowledge of VPN, Virtual Private LAN Service (VPLS), and Multiprotocol Layer Switching (MPLS) technologies.

Restrictions for BGP Support for the L2VPN Address Family

- For route maps used within BGP, all commands related to prefix processing, tag processing, and automated tag processing are ignored when used in L2VPN address family configuration mode. All other route map commands are supported.
- BGP multipaths and confederations are not supported in the L2VPN address family.

Information About BGP Support for the L2VPN Address Family

L2VPN Address Family

In Cisco IOS Release 12.2(33)SRB and later releases, support for the L2VPN address family is introduced. L2VPN is defined as a secure network that operates inside an unsecured network by using an encryption technology such as IP security (IPsec) or generic routing encapsulation (GRE). The L2VPN address family is configured in BGP routing configuration mode, and within the L2VPN address family the VPLS subsequent address family identifier (SAFI) is supported.

BGP support for the L2VPN address family introduces a BGP-based autodiscovery mechanism to distribute L2VPN endpoint provisioning information. BGP uses a separate L2VPN Routing Information Base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 VFI is configured. Prefix and path information is stored in the L2VPN database, allowing BGP to make best-path decisions. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to set up a pseudowire mesh to support L2VPN-based services.

The BGP autodiscovery mechanism facilitates the setting up of L2VPN services, which are an integral part of the Cisco IOS Virtual Private LAN Service (VPLS) feature. VPLS enables flexibility in deploying services by connecting geographically dispersed sites as a large LAN over high-speed Ethernet in a robust and scalable IP MPLS network. For more details about VPLS, see the VPLS Autodiscovery: BGP Based feature.

In L2VPN address family, the following BGP commands are supported:

- **bgp nexthop**
- **bgp scan-time**
- **neighbor activate**
- **neighbor advertisement-interval**
- **neighbor allowas-in**
- **neighbor capability**
- **neighbor inherit**

- **neighbor maximum-prefix**
- **neighbor next-hop-self**
- **neighbor next-hop-unchanged**
- **neighbor peer-group**
- **neighbor remove-private-as**
- **neighbor route-map**
- **neighbor route-reflector-client**
- **neighbor send-community**
- **neighbor soo**
- **neighbor weight**

Note For route reflectors using L2VPNs, the **neighbor next-hop-self** and **neighbor next-hop-unchanged**commands are not supported.

For route maps used within BGP, all commands related to prefix processing, tag processing, and automated tag processing are ignored when used in L2VPN address family configuration. All other route map commands are supported.

BGP multipaths and confederations are not supported in the L2VPN address family.

VPLS ID

A VPLS ID is a BGP extended community value that identifies the VPLS domain. Manual configuration of this ID is optional because a default VPLS ID is generated using the BGP autonomous system number and the configured VPN ID. A VPLS ID can be composed in one of two ways: with an autonomous system number and an arbitrary number or with an IP address and an arbitrary number.

You can enter a VPLS ID in either of these formats:

- Enter a 16-bit autonomous system number, a colon, and a 32-bit number. For example:

45000:3

- Enter a 32-bit IP address, a colon, and a 16-bit number. For example:

192.168.10.15:1

How to Configure BGP Support for the L2VPN Address Family

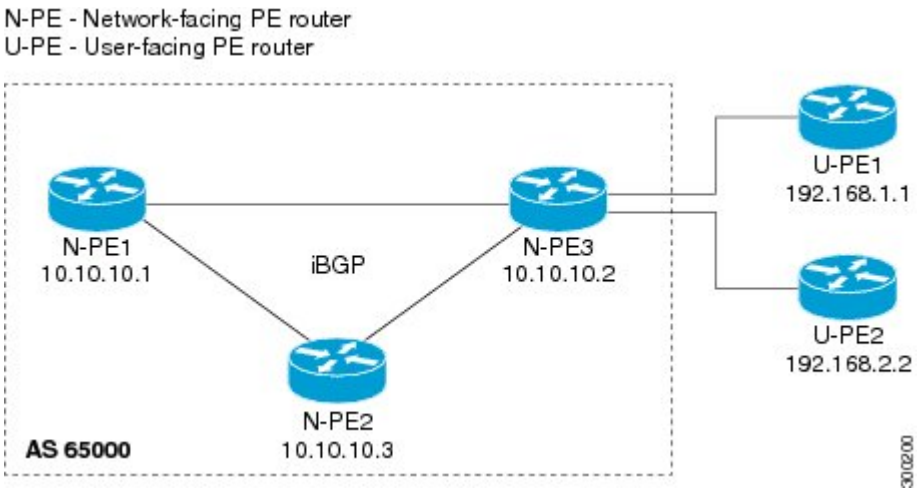
Configuring VPLS Autodiscovery Using BGP and the L2VPN Address Family

Perform this task to implement VPLS autodiscovery of each provider edge (PE) router that is a member of a specific VPLS. In Cisco IOS Release 12.2(33)SRB, the BGP L2VPN address family was introduced with a separate L2VPN RIB that contains endpoint provisioning information. BGP learns the endpoint provisioning information from the L2VPN database, which is updated each time any Layer 2 (L2) VFI is configured. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to set up a pseudowire mesh to support L2VPN-based services.

BGP-based VPLS autodiscovery eliminates the need to manually provision a VPLS neighbor. After a PE router configures itself to be a member of a particular VPLS, information needed to set up connections to remote routers in the same VPLS is distributed by a discovery process. When the discovery process is complete, each member of the VPLS will have the information needed to set up VPLS pseudowires to form the full mesh of pseudowires needed for the VPLS.

This task is configured at router N-PE3 in the figure below and must be repeated at routers N-PE1 and N-PE2 with the appropriate changes such as different IP addresses. For a full configuration of these routers, see "Configuring VPLS Autodiscovery Using BGP" and the "L2VPN Address Family Example".

Figure 8: Network Diagram for BGP Autodiscovery Using the L2VPN Address Family



In this task, the PE router N-PE3 in the figure above is configured with a Layer 2 router ID, a VPN ID, a VPLS ID, and is enabled to automatically discover other PE routers that are part of the same VPLS domain. A BGP session is created to activate BGP neighbors in the L2VPN address family. Finally, two optional **show** commands are entered to verify the steps in the task.

If a route reflector (RR) node is provisioned for a new virtual forwarding instance (VFI), BGP announces the entire current table from the L2VPN address family identifier (AFI) to the L2VPN xconnect database to make sure that the virtual circuits (VC) are active.

Before You Begin

This task assumes that MPLS is configured with VPLS options. For more details, see the VPLS Autodiscovery: BGP Based feature.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **l2 router-id** *ip-address*
4. **l2 vfi** *vfi-name* **autodiscovery**
5. **vpn id** *vpn-id*
6. **vpls-id** *vpls-id*
7. **exit**
8. Repeat Step 4 through Step 6 to configure other L2 VFIs and associated VPN and VPLS IDs.
9. **router bgp** *autonomous-system-number*
10. **no bgp default ipv4-unicast**
11. **bgp log-neighbor-changes**
12. **bgp update-delay** *seconds*
13. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number*
14. **neighbor** {*ip-address*| *peer-group-name*} **update-source** *interface-type interface-number*
15. Repeat Step 13 and Step 14 to configure other BGP neighbors.
16. **address-family l2vpn** [**vpls**]
17. **neighbor** {*ip-address* | *peer-group-name*} **activate**
18. **neighbor** {*ip-address*| *peer-group-name*} **send-community**[**both**| **standard**| **extended**]
19. Repeat Step 17 and Step 18 to activate other BGP neighbors in L2VPN address family.
20. **end**
21. **show vfi**
22. **show ip bgp l2vpn vpls** {**all** | **rd** *vpn-rd*}

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**l2 router-id** *ip-address* **Example:** `Router(config)# l2 router-id 10.1.1.3`	Specifies a router ID (in IP address format) for the PE router to use with VPLS autodiscovery pseudowires. • In this example, the L2 router ID is defined as 10.1.1.3.
Step 4	**l2 vfi** *vfi-name* **autodiscovery** **Example:** `Router(config)# l2 vfi customerA autodiscovery`	Creates an L2 VFI, enables the VPLS PE router to automatically discover other PE routers that are part of the same VPLS domain, and enters L2 VFI autodiscovery configuration mode. • In this example, the L2 VFI named customerA is created.
Step 5	**vpn id** *vpn-id* **Example:** `Router(config-vfi)# vpn id 100`	Specifies a VPN ID. • Use the same VPN ID for the PE routers that belong to the same VPN. Make sure that the VPN ID is unique for each VPN in the service provider network. • Use the *vpn-id* argument to specify a number in the range from 1 to 4294967295. • In this example, a VPN ID of 100 is specified.
Step 6	**vpls-id** *vpls-id* **Example:** `Router(config-vfi)# vpls-id 65000:100`	(Optional) Specifies a VPLS ID. • The VPLS ID is an identifier that is used to identify the VPLS domain. This command is optional because a default VPLS ID is automatically generated using the BGP autonomous system number and the VPN ID configured for the VFI. Only one VPLS ID can be configured per VFI, and the same VPLS ID cannot be configured in multiple VFIs on the same router. • In this example, a VPLS ID of 65000:100 is specified.
Step 7	**exit** **Example:** `Router(config-vfi)# exit`	Exits L2 VFI autodiscovery configuration mode and returns to global configuration mode.
Step 8	Repeat Step 4 through Step 6 to configure other L2 VFIs and associated VPN and VPLS IDs.	—
Step 9	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 10	**no bgp default ipv4-unicast**	Disables the IPv4 unicast address family for the BGP routing process.

	Command or Action	Purpose
	Example: `Router(config-router)# no bgp default ipv4-unicast`	**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 11	**bgp log-neighbor-changes** **Example:** `Router(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 12	**bgp update-delay** *seconds* **Example:** `Router(config-router)# bgp update-delay 1`	Sets the maximum initial delay period before a BGP-speaking networking device sends its first updates. • Use the *seconds* argument to set the delay period.
Step 13	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 10.10.10.1 remote-as 65000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 10.10.10.1 is an internal BGP neighbor.
Step 14	**neighbor** {*ip-address*| *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Router(config-router)# neighbor 10.10.10.1 update-source loopback 1`	(Optional) Configures a router to select a specific source or interface to receive routing table updates. • This example uses a loopback interface. The advantage to this configuration is that the loopback interface is not as susceptible to the effects of a flapping interface.
Step 15	Repeat Step 13 and Step 14 to configure other BGP neighbors.	--

	Command or Action	Purpose
Step 16	**address-family l2vpn** [**vpls**] **Example:** `Router(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 17	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Router(config-router-af)# neighbor 10.10.10.1 activate`	Enables the neighbor to exchange information for the L2VPN VPLS address family with the local router. **Note** If you have configured a BGP peer group as a neighbor, you do not use this step. BGP peer groups are activated when a BGP parameter is configured. For example, the **neighbor send-community** command in the next step will automatically activate a peer group.
Step 18	**neighbor** {*ip-address*| *peer-group-name*} **send-community**[**both**| **standard**| **extended**] **Example:** `Router(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 19	Repeat Step 17 and Step 18 to activate other BGP neighbors in L2VPN address family.	--
Step 20	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 21	**show vfi** **Example:** `Router# show vfi`	(Optional) Displays information about the configured VFI instances.
Step 22	**show ip bgp l2vpn vpls** {**all** | **rd** *vpn-rd*} **Example:** `Router# show ip bgp l2vpn vpls all`	(Optional) Displays information about the L2 VPN VPLS address family.

Examples

The following is sample output from the **show vfi** command that shows two VFIs, CustomerA and CustomerB, with their associated VPN and VPLS IDs:

```
Router# show vfi

Legend: RT=Route-target, S=Split-horizon, Y=Yes, N=No
VFI name: customerA, state: down, type: multipoint
  VPN ID: 100, VPLS-ID: 65000:100
  RD: 65000:100, RT: 65000:100
  Local attachment circuits:
  Neighbors connected via pseudowires:
  Peer Address     VC ID        Discovered Router ID    S
  10.10.10.1         100        10.10.10.99             Y
VFI name: customerB, state: down, type: multipoint
  VPN ID: 200, VPLS-ID: 65000:200
  RD: 65000:200, RT: 65000:200
  Local attachment circuits:
  Neighbors connected via pseudowires:
  Peer Address     VC ID        Discovered Router ID    S
  10.10.10.3         200                10.10.10.98    Y
```

The following is sample output from the **show ip bgp l2vpn vpls all** command that shows two VFIs identified by their VPN route distinguisher:

```
Router# show ip bgp l2vpn vpls all

BGP table version is 5, local router ID is 10.10.10.2
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
Route Distinguisher: 65000:100
*> 65000:100:10.10.10.1/96
                    0.0.0.0                              32768 ?
*>i65000:100:192.168.1.1/96
                    10.10.10.2               0    100      0 ?
Route Distinguisher: 65000:200
*> 65000:200:10.10.10.3/96
                    0.0.0.0                              32768 ?
*>i65000:200:192.168.2.2/96
                    10.10.10.2               0    100      0 ?
```

What to Do Next

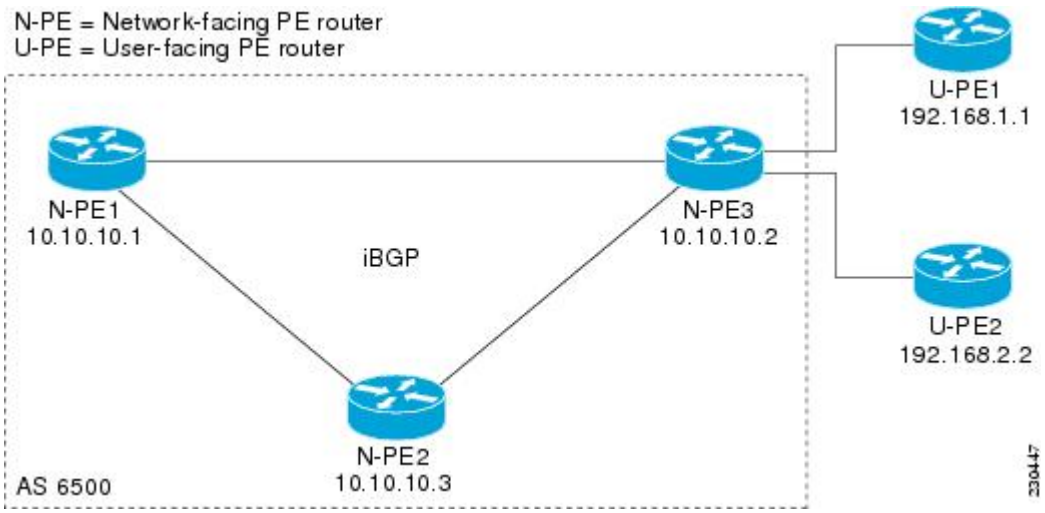
To configure more VPLS features, see the main VPLS documentation in the "VPLS Autodiscovery: BGP Based" module in the *MPLS Layer 2 VPNs Configuration Guide*.

Configuration Examples for BGP Support for the L2VPN Address Family

Example: Configuring VPLS Autodiscovery Using BGP and the L2VPN Address Family

In this configuration example, all the routers in autonomous system 65000 in the figure below are configured to provide BGP support for the L2VPN address family. VPLS autodiscovery is enabled and L2 VFI and VPN IDs are configured. BGP neighbors are configured and activated in L2VPN address family to ensure that the VPLS endpoint provisioning information is saved to a separate L2VPN RIB and then distributed to the other BGP peers in BGP update messages. When the endpoint information is received by the BGP peers, a pseudowire mesh is set up to support L2VPN-based services.

Figure 9: Network Diagram for VPLS Autodiscovery Using BGP and the L2VPN Address Family



Router N-PE1

```
ip subnet-zero
ip cef
no ip dhcp use vrf connected
!
no mpls traffic-eng auto-bw timers frequency 0
mpls label range 1000 2000
mpls label protocol ldp
l2 router-id 10.1.1.1
l2 vfi auto autodiscovery
 vpn id 100
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.1 255.255.255.255
!
interface Ethernet0/0
```

```
 description Backbone interface
 ip address 10.0.0.1 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.10.1.0 0.0.0.255 area 0
 network 192.168.0.0 0.0.0.255 area 0
!
router bgp 65000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.10.10.2 remote-as 65000
 neighbor 10.10.10.2 update-source Loopback 1
 neighbor 10.10.10.3 remote-as 65000
 neighbor 10.10.10.3 update-source Loopback 1
!
 address-family l2vpn vpls
 neighbor 10.10.10.2 activate
 neighbor 10.10.10.2 send-community extended
 neighbor 10.10.10.3 activate
 neighbor 10.10.10.3 send-community extended
 exit-address-family
!
ip classless
```

Router N-PE2

```
ip subnet-zero
ip cef
no ip dhcp use vrf connected
!
no mpls traffic-eng auto-bw timers frequency 0
mpls label range 2000 3000
mpls label protocol ldp
l2 router-id 10.1.1.2
l2 vfi auto autodiscovery
 vpn id 100
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.2 255.255.255.255
!
interface Ethernet0/0
 description Backbone interface
 ip address 10.0.0.2 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.10.1.0 0.0.0.255 area 0
 network 192.168.0.0 0.0.0.255 area 0
!
router bgp 65000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.10.10.1 remote-as 65000
 neighbor 10.10.10.1 update-source Loopback 1
 neighbor 10.10.10.3 remote-as 65000
 neighbor 10.10.10.3 update-source Loopback 1
!
 address-family l2vpn vpls
 neighbor 10.10.10.1 activate
 neighbor 10.10.10.1 send-community extended
 neighbor 10.10.10.3 activate
 neighbor 10.10.10.3 send-community extended
 exit-address-family
```

```
!
ip classless
```

Router N-PE3

```
ip subnet-zero
ip cef
no ip dhcp use vrf connected
!
no mpls traffic-eng auto-bw timers frequency 0
mpls label range 2000 3000
mpls label protocol ldp
l2 router-id 10.1.1.3
l2 vfi auto autodiscovery
 vpn id 100
!
pseudowire-class mpls
 encapsulation mpls
!
interface Loopback1
 ip address 10.1.1.3 255.255.255.255
!
interface Ethernet0/0
 description Backbone interface
 ip address 10.0.0.3 255.255.255.0
 mpls ip
!
router ospf 1
 log-adjacency-changes
 network 10.10.1.0 0.0.0.255 area 0
 network 192.168.0.0 0.0.0.255 area 0
!
router bgp 65000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 bgp update-delay 1
 neighbor 10.10.10.1 remote-as 65000
 neighbor 10.10.10.1 update-source Loopback 1
 neighbor 10.10.10.2 remote-as 65000
 neighbor 10.10.10.2 update-source Loopback 1
!
 address-family l2vpn vpls
 neighbor 10.10.10.1 activate
 neighbor 10.10.10.1 send-community extended
 neighbor 10.10.10.2 activate
 neighbor 10.10.10.2 send-community extended
 exit-address-family
!
ip classless
```

Where to Go Next

For more details about configuring VPLS autodiscovery, see the “VPLS Autodiscovery: BGP Based” module in the *MPLS Layer 2 VPNs Configuration Guide*.

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
VPLS Autodiscovery BGP Based	"VPLS Autodiscovery BGP Based" module in the *MPLS Layer 2 VPNS Configuration Guide*

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIB	MIBs Link
—	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

RFCs

RFC	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

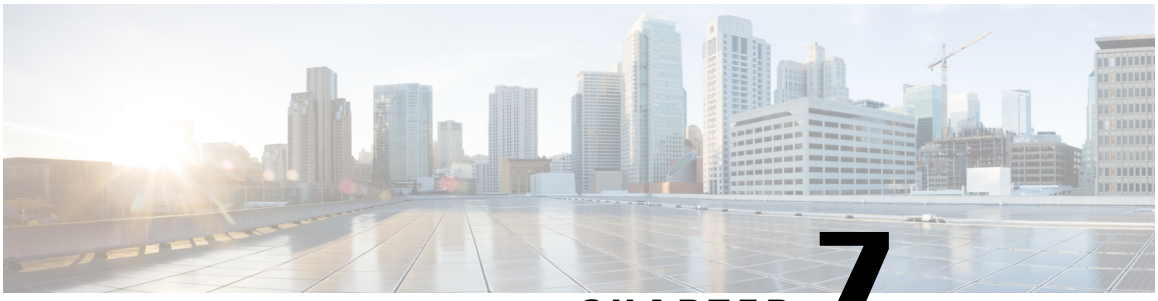
Feature Information for BGP Support for the L2VPN Address Family

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 9: Feature Information for BGP Support for the L2VPN Address Family

Feature Name	Releases	Feature Information
BGP Support for the L2VPN Address Family	15.1(1)SG	BGP support for the L2VPN address family introduces a BGP-based autodiscovery mechanism to distribute L2VPN endpoint provisioning information. BGP uses a separate L2VPN RIB to store endpoint provisioning information, which is updated each time any Layer 2 VFI is configured. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, the endpoint information is used to set up a pseudowire mesh to support L2VPN-based services. The following commands were introduced or modified by this feature: **address-family l2vpn**, **clear ip bgp l2vpn**, **show ip bgp l2vpn**.



CHAPTER 7

BGP Event-Based VPN Import

The BGP Event-Based VPN Import feature introduces a modification to the existing Border Gateway Protocol (BGP) path import process. The enhanced BGP path import is driven by events; when a BGP path changes, all of its imported copies are updated as soon as processing is available. Convergence times are significantly reduced because there is no longer any delay in the propagation of routes due to the software waiting for a periodic scanner time interval before processing the updates. To implement the new processing, new command-line interface (CLI) commands are introduced.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for BGP Event-Based VPN Import

Cisco Express Forwarding or distributed Cisco Express Forwarding must be enabled on all participating routers.

Information About BGP Event-Based VPN Import

BGP Event-Based VPN Import

The BGP Event-Based VPN Import feature introduces a modification to the existing BGP path import process. BGP Virtual Private Network (VPN) import provides importing functionality for BGP paths where BGP paths are imported from the BGP VPN table into a BGP virtual routing and forwarding (VRF) topology. In the existing path import process, when path updates occur, the import updates are processed during the next scan time which is a configurable interval of 5 to 15 seconds. The scan time adds a delay in the propagation of routes. The enhanced BGP path import is driven by events; when a BGP path changes, all of its imported copies are updated as soon as processing is available.

Using the BGP Event-Based VPN Import feature, convergence times are significantly reduced because provider edge (PE) routers can propagate VPN paths to customer edge (CE) routers without the scan time delay. Configuration changes such as adding imported route-targets to a VRF are not processed immediately, and are still handled during the 60-second periodic scanner pass.

Import Path Selection Policy

The BGP Event-Based VPN Import feature introduces three path selection policies:

- All—Import all available paths from the exporting net that match any Route Target (RT) associated with the importing VRF instance.
- Bestpath—Import the best available path that matches the RT of the VRF instance. If the bestpath in the exporting net does not match the RT of the VRF instance, a best available path that matches the RT of the VRF instance is imported.
- Multipath—Import the bestpath and all paths marked as multipaths that match the RT of the VRF instance. If there are no bestpath or multipath matches, then the best available path is selected.

Multipath and bestpath options can be restricted using an optional keyword to ensure that the selection is made only on the configured option. If the **strict** keyword is configured, the software disables the fall back safety option of choosing the best available path. If there are no paths appropriate to the configured option (bestpath or multipath) in the exporting net that match the RT of the VRF instance, then no paths are imported. This behavior matches the behavior of the software before the BGP Event-Based VPN Import feature was introduced.

When the restriction is not set, paths that are imported as the best available path are tagged. In **show** command output these paths are identified with the wording, “imported safety path.”

The paths existing in an exporting net that are considered for import into a VRF instance may have been received from another peer router and were not subject to the VPN importing rules. These paths may contain the same route-distinguisher (RD) information because the RD information is local to a router, but some of these paths do not match the RT of the importing VRF instance and are marked as “not-in-vrf” in the **show** command output. Any path that is marked as “not-in-vrf” is not considered as a bestpath because paths not in the VRF appear less attractive than paths in the VRF.

Import Path Limit

To control the memory utilization, a maximum limit of the number of paths imported from an exporting net can be specified per importing net. When a selection is made of paths to be imported from one or more exporting net, the first selection priority is a best path, the next selection priority is for multipaths, and the lowest selection priority is for nonmultipaths.

How to Configure BGP Event-Based VPN Import

Configuring a Multiprotocol VRF

Perform this task to configure a multiprotocol VRF that allows 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. In this task, only the IPv4 address family is configured, but we recommend using the multiprotocol VRF configuration for all new VRF configurations.

Note This task is not specific to the BGP Event-Based VPN Import feature.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf-name*
4. **rd** *route-distinguisher*
5. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
6. **address-family ipv4** [**unicast**]
7. **exit-address-family**
8. **exit**
9. **interface** *type number*
10. **vrf forwarding** *vrf-name*
11. **ip address** *ip-address mask*
12. **no shutdown**
13. **exit**
14. Repeat Step 3 through Step 13 to bind other VRF instances with an interface.
15. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Router(config)# vrf definition vrf-A`	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 4	**rd** *route-distinguisher* **Example:** `Router(config-vrf)# rd 45000:1`	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 5	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Router(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 both import routing information from, 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 6	**address-family ipv4** [**unicast**] **Example:** `Router(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters VRF address family configuration mode. • This step is required here to specify an address family for the VRF defined in the previous steps.
Step 7	**exit-address-family** **Example:** `Router(config-vrf-af)# exit-address-family`	Exits VRF address family configuration mode and returns to VRF configuration mode.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Router(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 9	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 1/1`	Enters interface configuration mode.
Step 10	**vrf forwarding** *vrf-name* **Example:** `Router(config-if)# vrf forwarding vrf-A`	Associates a VRF instance with the interface configured in Step 9. • When the interface is bound to a VRF, previously configured IP addresses are removed, and the interface is disabled.
Step 11	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.4.8.149 255.255.255.0`	Configures an IP address for the interface.
Step 12	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Restarts a disabled interface.
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 14	Repeat Step 3 through Step 13 to bind other VRF instances with an interface.	--
Step 15	**end** **Example:** `Router(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring Event-Based VPN Import Processing for BGP Paths

Perform this task to reduce convergence times when BGP paths change by configuring event-based processing for importing BGP paths into a VRF table. Two new CLI commands allow the configuration of a maximum number of import paths per importing net and the configuration of a path selection policy.

Before You Begin

This task assumes that you have previously configured the VRF to be used with the VRF address family syntax. To configure a VRF, see the "Configuring a Multiprotocol VRF" section earlier in this module.

Complete BGP neighbor configuration is also assumed. For an example configuration, see the "Example: Configuring Event-Based VPN Import Processing for BGP Paths" section in this module.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4 vrf** *vrf-name*
5. **import path selection** {**all** | **bestpath** [**strict**] | **multipath** [**strict**]}
6. **import path limit** *number-of-import-paths*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**address-family ipv4 vrf** *vrf-name* **Example:** `Router(config-router)# address-family ipv4 vrf vrf-A`	Specifies the IPv4 address family and enters address family configuration mode. • 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 5	**import path selection** {**all** | **bestpath** [**strict**] | **multipath** [**strict**]} **Example:** `Router(config-router-af)# import path selection all`	Specifies the BGP path selection policy for importing routes into a VRF table. • In this example, all paths that match any RT of the VRF instance are imported.
Step 6	**import path limit** *number-of-import-paths* **Example:** `Router(config-router-af)# import path limit 3`	Specifies, per importing net, a maximum number of BGP paths that can be imported from an exporting net.
Step 7	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Monitoring and Troubleshooting BGP Event-Based VPN Import Processing

Perform the steps in this task as required to monitor and troubleshoot the BGP event-based VPN import processing.

Only partial command syntax for the **show** commands used in this task is displayed. For more details, see the *Cisco IOS IP Routing: BGP Command Referenc*e.

SUMMARY STEPS

1. **enable**
2. **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} [*network-address* [*mask*]]
3. **show ip route** [**vrf** *vrf-name*] [*ip-address* [*mask*]]
4. **debug ip bgp vpnv4 unicast import** {**events** | **updates** [*access-list*]}

DETAILED STEPS

Step 1 **enable**
Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} [*network-address* [*mask*]]
In this example output, a safe import path selection policy is in effect because the **strict** keyword is not configured using the **import path selection** command. When a path is imported as the best available path (when the bestpath or multipaths are not eligible for import), the path is marked with "imported safety path," as shown in the output.

Example:

```
Router# show ip bgp vpnv4 all 172.17.0.0

BGP routing table entry for 45000:1:172.17.0.0/16, version 10
Paths: (1 available, best #1, table vrf-A)
Flag: 0x820
   Not advertised to any peer
   2, imported safety path from 50000:2:172.17.0.0/16
     10.0.101.1 from 10.0.101.1 (10.0.101.1)
       Origin IGP, metric 200, localpref 100, valid, internal, best
       Extended Community: RT:45000:100
```

The paths existing in an exporting net that are considered for import into a VRF instance may have been received from another peer router and were not subject to the VPN importing rules. These paths may contain the same route-distinguisher (RD) information because the RD information is local to a router, but some of these paths do not match the RT of the importing VRF instance and are marked as "not-in-vrf" in the **show** command output.

In the following example output, a path was received from another peer router and was not subject to the VPN importing rules. This path, 10.0.101.2, was added to the VPNv4 table and associated with the vrf-A net because it contains a match of the RD information although the RD information was from the original router. This path is not, however, an RT match for vrf-A and is marked as "not-in-vrf." Note that on the net for vrf-A, this path is not the bestpath because any paths that are not in the VRF appear less attractive than paths in the VRF.

Example:

```
Router# show ip bgp vpnv4 all 172.17.0.0

BBGP routing table entry for 45000:1:172.17.0.0/16, version 11
Paths: (2 available, best #2, table vrf-A)
Flag: 0x820
   Not advertised to any peer
   2
     10.0.101.2 from 10.0.101.2 (10.0.101.2)
       Origin IGP, metric 100, localpref 100, valid, internal, not-in-vrf
       Extended Community: RT:45000:200
       mpls labels in/out nolabel/16
   2
     10.0.101.1 from 10.0.101.1 (10.0.101.1)
       Origin IGP, metric 50, localpref 100, valid, internal, best
       Extended Community: RT:45000:100
       mpls labels in/out nolabel/16
```

Step 3 **show ip route** [**vrf** *vrf-name*] [*ip-address* [*mask*]]

In this example output, information about the routing table for VRF vrf-A is displayed:

Example:

```
Router# show ip route vrf vrf-A 172.17.0.0

Routing Table: vrf-A
Routing entry for 172.17.0.0/16
  Known via "bgp 1", distance 200, metric 50
  Tag 2, type internal
  Last update from 10.0.101.33 00:00:32 ago
  Routing Descriptor Blocks:
  * 10.0.101.33 (default), from 10.0.101.33, 00:00:32 ago
      Route metric is 50, traffic share count is 1
      AS Hops 1
      Route tag 2
      MPLS label: 16
      MPLS Flags: MPLS Required
```

Step 4 **debug ip bgp vpnv4 unicast import** {**events** | **updates** [*access-list*]}

Use this command to display debugging information related to the importing of BGP paths into a VRF instance table. The actual output depends on the commands that are subsequently entered.

Note If no access list to filter prefixes is specified when using the updates keyword, all updates for all prefixes are displayed and this may slow down your network.

Example:

```
Router# debug ip bgp vpnv4 unicast import events

BGP import events debugging is on
```

Configuration Examples for BGP Event-Based VPN Import

Example: Configuring Event-Based VPN Import Processing for BGP Paths

In this example, a VRF (vrf-A) is configured and VRF forwarding is applied to Fast Ethernet interface 1/1. In address family mode, the import path selection is set to all and the number of import paths is set to 3. Two BGP neighbors are configured under the IPv4 address family and activated under the VPNv4 address family.

```
vrf definition vrf-A
 rd 45000:1
 route-target import 45000:100
 address-family ipv4
  exit-address-family
!
interface FastEthernet1/1
 no ip address
 vrf forwarding vrf-A
 ip address 10.4.8.149 255.255.255.0
 no shut
 exit
!
router bgp 45000
 network 172.17.1.0 mask 255.255.255.0
```

```
neighbor 192.168.1.2 remote-as 40000
neighbor 192.168.3.2 remote-as 50000
address-family ipv4 vrf vrf-A
 import path selection all
 import path limit 3
 exit-address-family
address-family vpnv4
 neighbor 192.168.1.2 activate
 neighbor 192.168.3.2 activate
 end
```

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 2918	*Route Refresh Capability 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

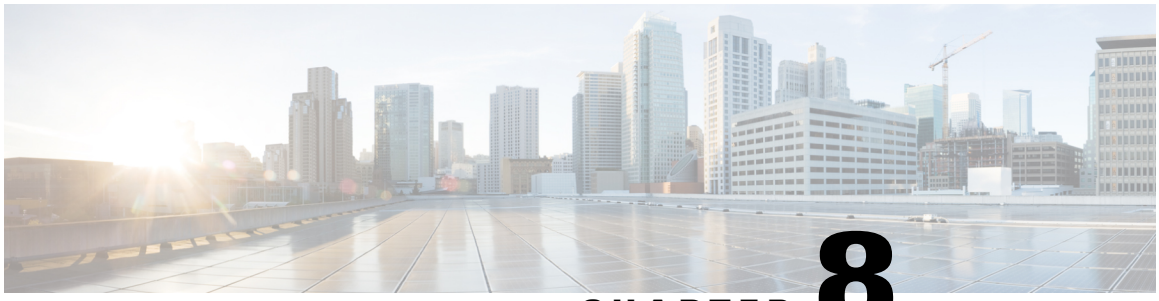
Feature Information for BGP Event-Based VPN Import

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 10: Feature Information for BGP Event-Based VPN Import

Feature Name	Releases	Feature Information
BGP Event-Based VPN Import	15.1(1)SG	The BGP Event-Based VPN Import feature introduces a modification to the existing Border Gateway Protocol (BGP) path import process. The enhanced BGP path import is driven by events; when a BGP path changes, all of its imported copies are updated as soon as processing is available. Convergence times are significantly reduced because there is no longer any delay in the propagation of routes due to the software waiting for a periodic scanner time interval before processing the updates. To implement the new processing, new command-line interface (CLI) commands are introduced. The following commands were introduced or modified: **bgp scan-time**, **import path limit**, **import path selection**, **maximum-paths eibgp**, **maximum-paths ibgp**, **show ip bgp vpnv4**.



CHAPTER 8

BGP Support for BFD

Bidirectional Forwarding Detection (BFD) is a detection protocol 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 will be easier, and reconvergence time will be consistent and predictable. The main benefit of implementing BFD for BGP is a significantly faster reconvergence time.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Support for BFD

BFD for BGP

Bidirectional Forwarding Detection (BFD) is a detection protocol 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 will be easier, and reconvergence time will be consistent and predictable. The main benefit of implementing BFD for BGP is a marked decrease in reconvergence time.

Caution BFD and BGP Graceful Restart capability cannot both be configured on a router running BGP. If an interface goes down, BFD detects the failure and indicates that the interface cannot be used for traffic forwarding and the BGP session goes down, but graceful restart still allows traffic forwarding on platforms that support NSF even though the BGP session is down, allowing traffic forwarding using the interface that is down. Configuring both BFD and BGP graceful restart for NSF on a router running BGP may result in suboptimal routing.

See also the "Configuring BGP Neighbor Session Options" chapter, the section "Configuring BFD for BGP IPv6 Neighbors."

For more details about BFD, see the *Cisco IOS IP Routing: BFD Configuration Guide*.

How to Decrease BGP Convergence Time Using BFD

Prerequisites

- Cisco Express Forwarding (CEF) and IP routing must be enabled on all participating routers.
- BGP must be configured on the routers before BFD is deployed. You should implement fast convergence for the routing protocol that you are using. See the IP routing documentation for your version of Cisco IOS software for information on configuring fast convergence.

Restrictions

- For the Cisco implementation of BFD Support for BGP in Cisco IOS Release15.1(1)SG, only asynchronous mode is supported. In asynchronous mode, either BFD peer can initiate a BFD session.
- IPv6 encapsulation is supported.
- BFD works only for directly-connected neighbors. BFD neighbors must be no more than one IP hop away. Multihop configurations are not supported.
- Configuring both BFD and BGP Graceful Restart for NSF on a router running BGP may result in suboptimal routing.

Decreasing BGP Convergence Time Using BFD

You start a BFD process by configuring BFD on the interface. When the BFD process is started, no entries are created in the adjacency database, in other words, no BFD control packets are sent or received. The adjacency creation takes places once you have configured BFD support for the applicable routing protocols.

The first two tasks must be configured to implement BFD support for BGP to reduce the BGP convergence time. The third task is an optional task to help monitor or troubleshoot BFD.

See also the "Configuring BFD for BGP IPv6 Neighbors" section in the "Configuring BGP Neighbor Session Options" module.

Configuring BFD Session Parameters on the Interface

The steps in this procedure show how to configure BFD on the interface by setting the baseline BFD session parameters on an interface. Repeat the steps in this procedure for each interface over which you want to run BFD sessions to BFD neighbors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface FastEthernet 6/0`	Enters interface configuration mode.
Step 4	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier* **Example:** `Router(config-if)# bfd interval 50 min_rx 50 multiplier 5`	Enables BFD on the interface.

	Command or Action	Purpose
Step 5	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode.

Configuring BFD Support for BGP

Perform this task to configure BFD support for BGP, so that BGP is a registered protocol with BFD and will receive forwarding path detection failure messages from BFD.

Before You Begin

- BGP must be running on all participating routers.
- The baseline parameters for BFD sessions on the interfaces over which you want to run BFD sessions to BFD neighbors must be configured. See "Configuring BFD Session Parameters on the Interface" for more information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **fall-over bfd**
5. **end**
6. **show bfd neighbors** [**details**]
7. **show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp tag1`	Specifies a BGP process and enters router configuration mode.
Step 4	**neighbor** *ip-address* **fall-over bfd** **Example:** `Router(config-router)# neighbor 172.16.10.2 fall-over bfd`	Enables BFD support for fallover.
Step 5	**end** **Example:** `Router(config-router)# end`	Returns the router to privileged EXEC mode.
Step 6	**show bfd neighbors** [**details**] **Example:** `Router# show bfd neighbors detail`	Verifies that the BFD neighbor is active and displays the routing protocols that BFD has registered.
Step 7	**show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]] **Example:** `Router# show ip bgp neighbors`	Displays information about BGP and TCP connections to neighbors.

Monitoring and Troubleshooting BFD

To monitor or troubleshoot BFD, perform one or more of the steps in this section.

SUMMARY STEPS

1. **enable**
2. **show bfd neighbors** [**details**]
3. **debug bfd** [**event** | **packet** | **ipc-error** | **ipc-event** | **oir-error** | **oir-event**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show bfd neighbors** [**details**] **Example:** `Router# show bfd neighbors details`	(Optional) Displays the BFD adjacency database. • The **details** keyword shows all BFD protocol parameters and timers per neighbor.
Step 3	**debug bfd** [**event** | **packet** | **ipc-error** | **ipc-event** | **oir-error** | **oir-event**] **Example:** `Router# debug bfd packet`	(Optional) Displays debugging information about BFD packets.

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
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

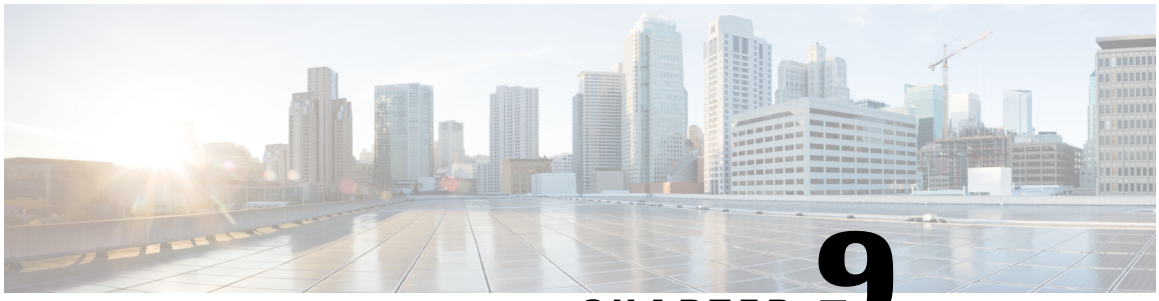
Feature Information for BGP Support for BFD

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 11: Feature Information for BGP Support for BFD

Feature Name	Releases	Feature Information
BGP Support for BFD	15.2(1)E	Bidirectional Forwarding Detection (BFD) is a detection protocol 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 will be easier, and reconvergence time will be consistent and predictable. The main benefit of implementing BFD for BGP is a significantly faster reconvergence time. The following commands were introduced or modified by this feature: **bfd**, **neighbor fall-over**, **show bfd neighbors**, and **show ip bgp neighbors**.



CHAPTER 9

BGP VPLS Auto Discovery Support on Route Reflector

BGP Route Reflector was enhanced to be able to reflect BGP VPLS prefixes without having VPLS explicitly configured on the route reflector.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP VPLS Auto Discovery Support on Route Reflector

BGP VPLS Autodiscovery Support on Route Reflector

In Cisco IOS Release 12.2(33)SRE, BGP VPLS Autodiscovery Support on Route Reflector was introduced. On the Cisco 7600 and Cisco 7200 series routers, BGP Route Reflector was enhanced to be able to reflect BGP VPLS prefixes without having VPLS explicitly configured on the route reflector. The route reflector

reflects the VPLS prefixes to other provider edge (PE) routers so that the PEs do not need to have a full mesh of BGP sessions. The network administrator configures only the BGP VPLS address family on the route reflector.

For an example of a route reflector configuration that can reflect VPLS prefixes, see the "Example: BGP VPLS Autodiscovery Support on Route Reflector" section. For more information about VPLS Autodiscovery, see the "VPLS Autodiscovery BGP Based" module in the *MPLS Layer 2 VPNs Configuration Guide* .

Configuration Example for BGP VPLS Auto Discovery Support on Route Reflector

Example: BGP VPLS Autodiscovery Support on Route Reflector

In the following example, a host named PE-RR (indicating Provider Edge Route Reflector) is configured as a route reflector capable of reflecting VPLS prefixes. The VPLS address family is configured by **address-family l2vpn vpls** command.

```
hostname PE-RR
!
router bgp 1
 bgp router-id 1.1.1.3
 no bgp default route-target filter
 bgp log-neighbor-changes
neighbor iBGP_PEERS peer-group
neighbor iBGP_PEERS remote-as 1
neighbor iBGP_PEERS update-source Loopback1
neighbor 1.1.1.1 peer-group iBGP_PEERS
neighbor 1.1.1.2 peer-group iBGP_PEERS
!
address-family l2vpn vpls
  neighbor iBGP_PEERS send-community extended
  neighbor iBGP_PEERS route-reflector-client
  neighbor 1.1.1.1 peer-group iBGP_PEERS
  neighbor 1.1.1.2 peer-group iBGP_PEERS
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

Standards and RFCs

Standard/RFC	Title
RFC 2918	*Route Refresh Capability 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

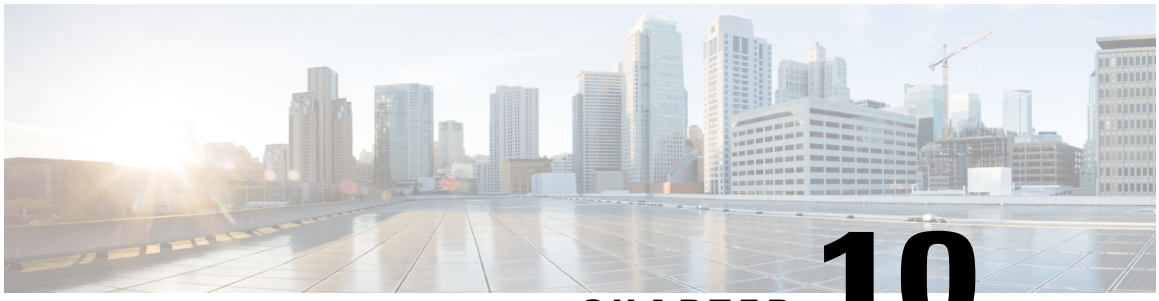
Feature Information for BGP VPLS Auto Discovery Support on Route Reflector

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 12: Feature Information for BGP VPLS Auto Discovery Support on Route Reflector

Feature Name	Releases	Feature Information
BGP VPLS Auto Discovery Support on Route Reflector	15.1(1)SG	BGP Route Reflector was enhanced to be able to reflect BGP VPLS prefixes without having VPLS explicitly configured on the route reflector.



CHAPTER 10

BGP Enhanced Route Refresh

The BGP Enhanced Route Refresh feature provides a way for Border Gateway Protocol (BGP) to find route inconsistencies, and in that unlikely event, to synchronize BGP peers without a hard reset. The feature is enabled by default; there are two optional timers.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information for BGP Enhanced Route Refresh, on page 117.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Information About BGP Enhanced Route Refresh

BGP Enhanced Route Refresh Functionality

During session establishment, BGP peers exchange with each other their capability to do the BGP Enhanced Route Refresh feature. The feature is enabled by default.

It is not expected that the peers will become inconsistent with each other. That might only happen in an extreme corner case, and if that happens, this feature helps to identify that and synchronize the peers without a hard reset.

If two peers are capable of Enhanced Route Refresh, each peer will generate a Route-Refresh Start-of-RIB (SOR) message before it advertises the Adj-RIB-Out, and will generate a Route-Refresh End-of-RIB (EOR) message after it advertises the Adj-RIB-Out. A BGP speaker receiving an EOR message from its peer removes the routes that were not re-advertised as part of Route Refresh response by the peer.

In the unlikely event the router has stale routes remaining after receiving the EOR message or after the EOR timer expires, that means the peers were not consistent with each other. This information can be used to check whether routes are consistent.

BGP Enhanced Route Refresh Timers

These timers need not be configured under normal circumstances.You could configure one or both timers if you observe there is continuous route flapping to the extent that a Route Refresh EOR cannot be generated.

The first timer applies to the router when it should be receiving the EOR message, but is not receiving one. The second timer applies to the router when it should be sending the EOR message.

- Stale path timer—If the **bgp refresh stalepath-time** command is configured and the router does not receive a Route-Refresh EOR message after an Adj-RIB-Out, the router removes the stale routes from the BGP table after the timer expires. The stale path timer is started when the router receives a Route-Refresh SOR message.
- Maximum EOR timer—If the **bgp refresh max-eor-time** command is configured and the router is unable to generate a Route-Refresh EOR message, a Route-Refresh EOR message is generated after the timer expires.

Both timers are configurable. By default, they are both disabled (set to 0 seconds).

Syslog Messages Generated by the BGP Enhanced Route Refresh

The following are examples of syslog messages that are generated when a peer deletes stale routes after receiving the Route-Refresh EOR message or after the stale path timer expires. The messages help you to know whether the routers were inconsistent.

```
Net 300:300:3.3.0.0/0 from bgp neighbor IPv4 MDT 10.0.101.1 is stale after refresh EOR
(rate-limited)
Net 300:300:3.3.0.0/0 from bgp neighbor IPv4 MDT 10.0.101.1 is stale after refresh stale-path
 timer expiry (rate-limited)
```

The following are examples of messages logged after a Route-Refresh EOR or after the stale path timer expires, which indicate the total number of stale paths that were from the neighbor.

```
3 stale-paths deleted from bgp neighbor IPv4 MDT 10.0.101.1 after refresh EOR
3 stale-paths deleted from bgp neighbor IPv4 MDT 10.0.101.1 after refresh stale-path timer
 expiry
```

How to Set Timers for BGP Enhanced Route Refresh

Set Timers for BGP Enhanced Route Refresh

The BGP Enhanced Route Refresh feature is enabled by default; the timers are disabled by default. Perform this task if you want to set the optional timers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system*
4. **bgp refresh stalepath-time** *seconds*
5. **bgp refresh max-eor-time** *seconds*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**bgp refresh stalepath-time** *seconds* **Example:** `Router(config-router)# bgp refresh stalepath-time 1200`	(Optional) Causes the router to remove stale routes from the BGP table after the timer expires, even if the router does not receive a Route-Refresh End-of-RIB message. • Valid values are from 600 to 3600, or 0. • The default is 0, meaning the stale-path timer is disabled. • The stale path timer is started when a router receives a Route-Refresh Start-of-RIB message.

	Command or Action	Purpose
Step 5	**bgp refresh max-eor-time** *seconds* **Example:** `Router(config-router)# bgp refresh max-eor-time 1200`	(Optional) Specifies that if BGP is unable to generate a Route-Refresh End-of-RIB (EOR) message, a Route-Refresh EOR is generated after the timer expires. • Valid values are from 600 to 3600, or 0. • The default is 0, meaning the max-eor timer is disabled.

Configuration Examples for BGP Enhanced Route Refresh

Example: Setting Timers for BGP Enhanced Route Refresh

In the following example, if no Route-Refresh EOR message is received after 800 seconds, stale routes will be removed from the BGP table. If no Route-Refresh EOR message is generated after 800 seconds, one is generated.

```
router bgp 65000
 bgp refresh stalepath-time 800
 bgp refresh max-eor-time 800
```

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

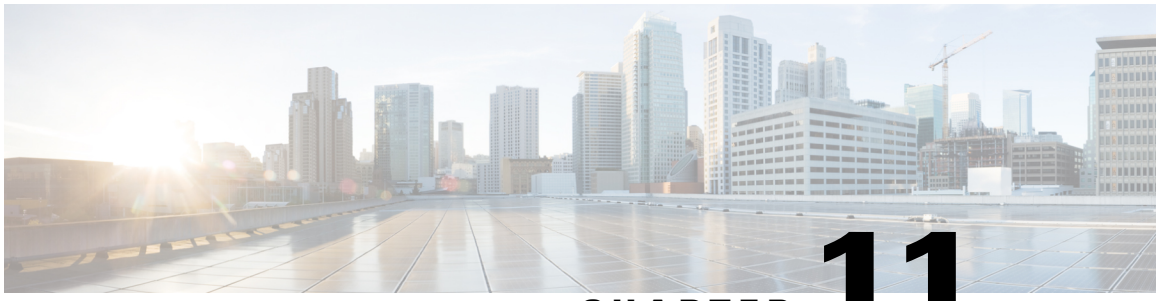
Feature Information for BGP Enhanced Route Refresh

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 13: Feature Information for BGP Enhanced Route Refresh

Feature Name	Releases	Feature Information
BGP Enhanced Route Refresh	15.1(2)SG	The BGP Enhanced Route Refresh feature provides a way for BGP to find route inconsistencies, and in that unlikely event, to synchronize BGP peers without a hard reset. The following commands were introduced: • **bgp refresh max-eor-time** • **bgp refresh stalepath-time**



CHAPTER 11

Configuring BGP Consistency Checker

The BGP Consistency Checker feature provides a way to identify certain types of BGP route inconsistencies with peers: next-hop label inconsistency, RIB-out inconsistency, and aggregation inconsistency. Upon finding such an inconsistency, the system sends a syslog error message and takes appropriate action if configured to do so.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Consistency Checker

BGP Consistency Checker

A BGP route inconsistency with a peer occurs when an update or a withdraw is not sent to a peer, and black-hole routing can result. To identify that issue, BGP consistency checker was created as a low-priority process that does nexthop-label, RIB-out, and aggregation consistency checks at a configurable interval. When enabled,

BGP consistency checker is performed for all address families. Configuring BGP consistency checker is recommended.

Once the process identifies such an inconsistency, it will report the inconsistency with a syslog message and optionally take action if the **auto-repair** keyword is specified. The action taken depends on the type of inconsistency found.

- Next-Hop Label Consistency Check—When two paths have the same next hop because they are advertised by the same provider edge router (PE), they should also have the same next-hop label. If the labels are different, there is an inconsistency. If the **auto-repair** keyword is specified, the system will send a route-refresh request.
- RIB-Out Consistency Check—If a network passes an outbound policy and is not sent, or if a network does not pass an outbound policy and is sent, there is an inconsistency. If the **auto-repair** keyword is specified, the system will send a route-refresh request.
- Aggregation Consistency Check—If specific routes and the aggregated route become out of sync, an inconsistency can occur. Either the **error-message** keyword or the **auto-repair** keyword will trigger aggregation reevaluation.

In the unlikely event that you receive a syslog message about an inconsistency, notify your Cisco technical support representative with the syslog message exactly as it appears. The following are examples of such syslog messages:

- "Net 10.0.0.0/32 has Nexthop-Label inconsistency."
- "Net 10.0.0.0/32 in IPv4 Unicast has rib-out inconsistency for update-group 4 - outbound-policy fails."

How to Configure BGP Consistency Checker

Configure BGP Consistency Checker

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp consistency-checker** {**error-message** | **auto-repair**} [**interval** *minutes*]
5. **end**
6. **show ip bgp** [**vpnv4** | **vpnv6**] **all inconsistency nexthop-label**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 500`	Configures a BGP routing process.
Step 4	**bgp consistency-checker** {**error-message** | **auto-repair**} [**interval** *minutes*] **Example:** `Router(config-router)# bgp consistency-checker auto-repair interval 720`	Enables BGP consistency checker. • The default interval is 1440 minutes (one day). The range is 5 to 1440 minutes.
Step 5	**end** **Example:** `Router(config-router)# end`	Ends the current configuration and returns to privileged EXEC mode.
Step 6	**show ip bgp** [**vpnv4** | **vpnv6**] **all inconsistency nexthop-label** **Example:** `Router# show ip bgp all inconsistency nexthop-label`	(Optional) Displays routes that have a nexthop-label inconsistency found. • This step is not part of configuring the feature; it is provided in case you receive a syslog message about a nexthop-label inconsistency and you want to display those routes.

Configuration Examples for BGP Consistency Checker

Example: Configuring BGP Consistency Checker

The following example configures BGP consistency checker with auto-repair at the default interval of one day:

```
router bgp 65000
 bgp consistency-checker auto-repair
```

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

MIBs

MIB	MIBs Link
—	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

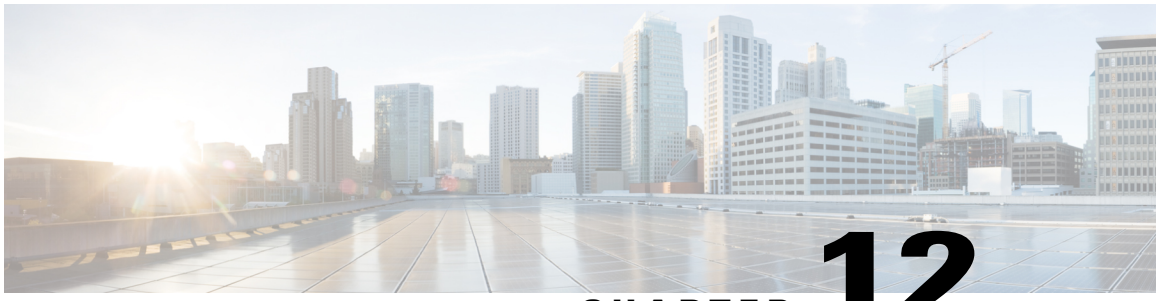
Feature Information for BGP Consistency Checker

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for BGP Consistency Checker

Feature Name	Releases	Feature Information
BGP Consistency Checker	15.1(2)SG	The BGP Consistency Checker feature provides a way to identify three types of BGP route inconsistencies with peers: next-hop label inconsistency, RIB-out inconsistency, and aggregation inconsistency. Upon finding such inconsistency, the system sends a syslog error message and takes appropriate action if configured to do so. The following command was introduced: **bgp consistency-checker** The following command was modified: **show ip bgp vpnv4**.



CHAPTER 12

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.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About 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 routers 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 routers. Therefore, routers 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 routers.

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.

BGP GSHUT Enhancement

The BGP Graceful Shutdown (GSHUT) Enhancement feature enables graceful shutdown of either all neighbors or only virtual routing and forwarding (VRF) neighbors across BGP sessions. To enable the BGP GSHUT enhancement feature on the device, you must configure either the **community** keyword or the **local-preference** keyword in the **bgp graceful-shutdown all** command. Use the **activate** keyword to activate graceful shutdown either across all neighbors or only across all VRF neighbors, across all BGP sessions.

How to Configure BGP Graceful Shutdown

Shutting Down a BGP Link Gracefully

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ipv4-address* | *ipv6-address*} **remote-as** *number*
5. **neighbor** {*ipv4-address* | *ipv6-address* | *peer-group-name*} **shutdown graceful** *seconds* {**community** *value* [**local-preference** *value*] | **local-preference** *value*}
6. **end**
7. **show ip bgp community gshut**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**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 router 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. • 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 router 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.

	Command or Action	Purpose
		• 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.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ipv4-address* | *ipv6-address*} **remote-as** *number*
5. **neighbor** {*ipv4-address* | *ipv6-address*} **activate**
6. **neighbor** {*ipv4-address* | *ipv6-address*} **send-community**
7. **exit**
8. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
9. **match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}
10. **exit**
11. **ip community-list** {*standard* | **standard** *list-name*} {**deny** | **permit**} **gshut**
12. **router bgp** *autonomous-system-number*
13. **neighbor** *address* **route-map** *map-name* **in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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.

Configuring BGP GSHUT Enhancement

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp graceful-shutdown all** {**neighbors** | **vrfs**} *shutdown-time* {**community** *community-value* [**local-preference** *local-pref-value*] | **local-preference** *local-pref-value* [**community** *community-value*]}
5. **bgp graceful-shutdown all** {**neighbors** | **vrfs**} **activate**
6. **end**
7. **show ip bgp**
8. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**bgp graceful-shutdown all** {**neighbors**	**vrfs**} *shutdown-time* {**community** *community-value* [**local-preference** *local-pref-value*]	**local-preference** *local-pref-value* [**community** *community-value*]} **Example:** `Device(config-router)# bgp graceful-shutdown all neighbors 180 local-preference 20 community 10`	Enables the BGP GSHUT enhancement feature on the device.

	Command or Action	Purpose
Step 5	**bgp graceful-shutdown all** {**neighbors** | **vrfs**} **activate** **Example:** `Device(config-router)# bgp graceful-shutdown all neighbors activate`	Activates graceful shutdown across all neighbors or only across VRF neighbors for BGP sessions.
Step 6	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 7	**show ip bgp** **Example:** `Device# show ip bgp neighbors 10.2.2.2 | include shutdown`	Displays entries in the BGP routing table.
Step 8	**show running-config** **Example:** `Device# show running-config | session router bgp`	Displays running configuration on the device.

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
```

Example: BGP GSHUT Enhancement

The following example shows how to enable and activate the BGP GSHUT enhancement feature across all neighbors. In this example, the neighbors are configured to gracefully shutdown within the specified duration of 180 seconds.

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# bgp graceful-shutdown all neighbors 180 local-preference 20 community
 10
Device(config-router)# bgp graceful-shutdown all neighbors activate
Device(config-router)# end
```

Following is sample output from the **show ip bgp** command, which displays the graceful shutdown time for each neighbor. In this example, there are two IPv4 neighbors configured with IP address 10.2.2.2 and 172.16.2.1 and one VRF neighbor, tagged v1, is configured with IP address 192.168.1.1.

```
Device# show ip bgp neighbors 10.2.2.2 | include shutdown
```

```
Graceful Shutdown Timer running, schedule to reset the peer in 00:02:47 seconds
Graceful Shutdown Localpref set to 20
Graceful Shutdown Community set to 10

Device# show ip bgp neighbors 172.16.2.1 | include shutdown

Graceful Shutdown Timer running, schedule to reset the peer in 00:02:38 seconds
Graceful Shutdown Localpref set to 20
Graceful Shutdown Community set to 10

Device# show ip bgp vpnv4 vrf v1 neighbors 192.168.1.1 | include shutdown

Graceful Shutdown Timer running, schedule to reset the peer in 00:01:45 seconds
Graceful Shutdown Localpref set to 20
Graceful Shutdown Community set to 10
```

Following is sample output from the **show running-config** command, which displays information associated with the BGP session in router configuration mode:

```
Device# show running-config | session router bgp

router bgp 65000
bgp log-neighbor-changes
bgp graceful-shutdown all neighbors 180 local-preference 20 community 10
network 10.1.1.0 mask 255.255.255.0
neighbor 10.2.2.2 remote-as 40
neighbor 10.2.2.2 shutdown
neighbor 172.16.2.1 remote-as 10
neighbor 172.16.2.1 shutdown
!
address-family vpnv4
neighbor 172.16.2.1 activate
neighbor 172.16.2.1 send-community both
exit-address-family
!
address-family ipv4 vrf v1
neighbor 192.168.1.1 remote-as 30
neighbor 192.168.1.1 shutdown
neighbor 192.168.1.1 activate
neighbor 192.168.1.1 send-community both
exit-address-family
```

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

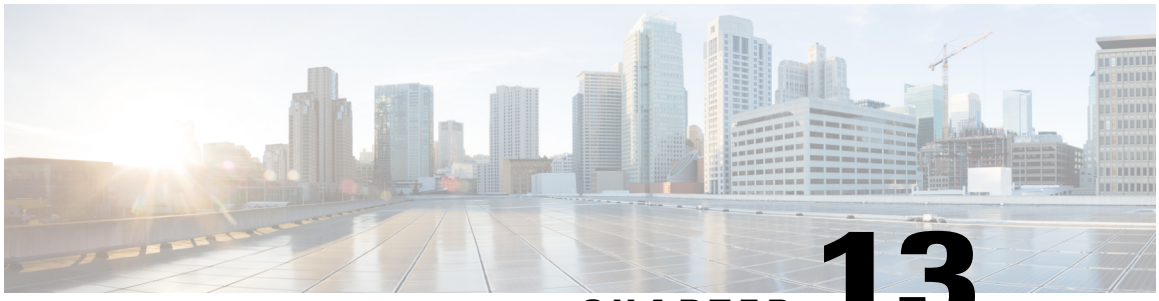
Feature Information for BGP Graceful Shutdown

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15: Feature Information for BGP Graceful Shutdown

Feature Name	Releases	Feature Information
BGP Graceful Shutdown	15.2(1)E	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. The following commands were modified: **ip community-list**, **neighbor shutdown**, **show ip bgp community**, and **show ip bgp vpnv4**.
BGP GSHUT Enhancement		The BGP Graceful Shutdown (GSHUT) Enhancement feature enables graceful shutdown of either all neighbors or only virtual routing and forwarding (VRF) neighbors across BGP sessions. The following command was introduced: **bgp graceful-shutdown all**.



CHAPTER 13

BGP Additional Paths

The BGP Additional Paths feature allows the advertisement of multiple paths through the same peering session for the same prefix without the new paths implicitly replacing any previous paths. This behavior promotes path diversity and reduces multi-exit discriminator (MED) oscillations.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Additional Paths

Problem That Additional Paths Can Solve

BGP routers and route reflectors (RRs) propagate only their best path over their sessions. The advertisement of a prefix replaces the previous announcement of that prefix (this behavior is known as an implicit withdraw). The implicit withdraw can achieve better scaling, but at the cost of path diversity.

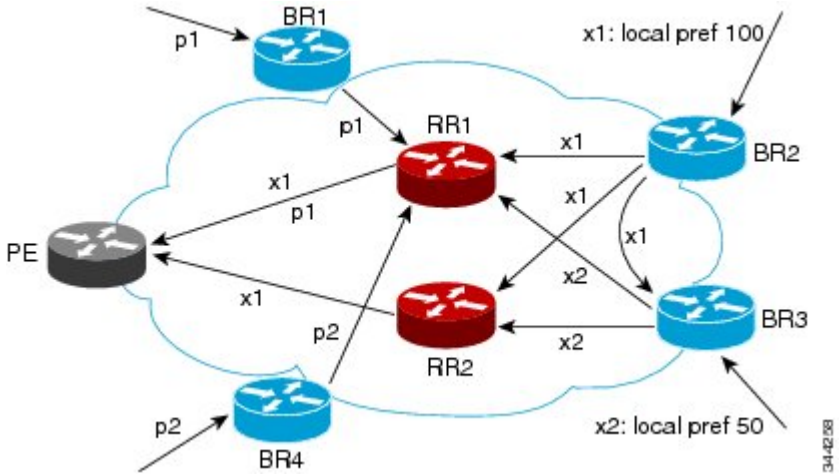
Path hiding can prevent efficient use of BGP multipath, prevent hitless planned maintenance, and can lead to MED oscillations and suboptimal hot-potato routing. Upon nexthop failures, path hiding also inhibits fast and local recovery because the network has to wait for BGP control plane convergence to restore traffic. The BGP Additional Paths feature provides a generic way of offering path diversity; the Best External or Best Internal features offer path diversity only in limited scenarios.

The BGP Additional Paths feature provides a way for multiple paths for the same prefix to be advertised without the new paths implicitly replacing the previous paths. Thus, path diversity is achieved instead of path hiding.

Path-Hiding Scenario

This section describes in more detail how path hiding can occur. In the following figure, we have prefix p with paths p1 and p2 advertised from BR1 and BR4 to RR1. RR1 selects the best path of the two and then advertises to PE only p1.

Figure 10: RR Hiding an Additional Path



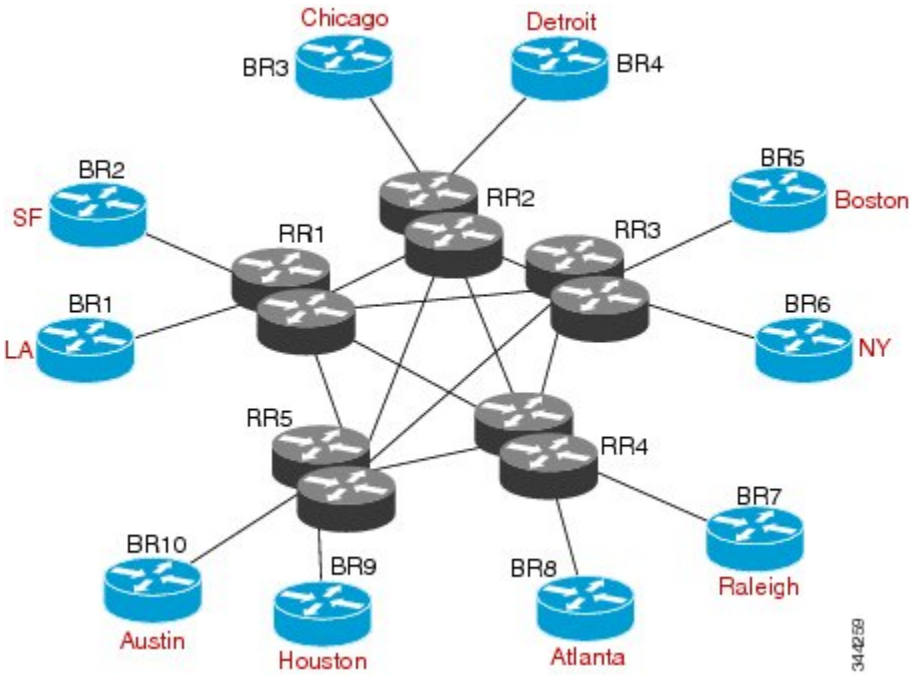
In the figure above, we also see prefix x with path x1 being advertised from BR2 to BR3 (which has path x2) with local preference 100. BR3 also has path x2, but due to routing policy, BR3 will advertise to the RRs x1 (not shown) instead of x2, and x2 will be suppressed. A user could enable the advertisement of best external on BR3 and thereby advertise x2 to the RRs, but, again, the RRs advertise only the best path.

Suboptimal Hot-Potato Routing Scenario

In order to minimize internal transport costs, transit ISPs try to forward packets to the closest exit point (according to Interior Gateway Protocol [IGP] cost). This behavior is known as hot-potato routing. In the distributed RR cluster model of the figure below, assume traffic coming from LA must go to Mexico. All links have the same IGP cost. If there are two exit points toward Mexico—one toward Austin and one toward Atlanta—the border router will try to send traffic to Austin based on the lower IGP cost from LA toward Austin than toward Atlanta. In a centralized RR model where the central RR resides where RR3 is (and RR1,

RR2, RR4, and RR5 do not exist), the closest exit point toward Mexico, as seen from RR3, might be Atlanta. Sending the traffic from LA toward Atlanta results in suboptimal hot-potato routing, which is not desirable.

Figure 11: Distributed RR Cluster

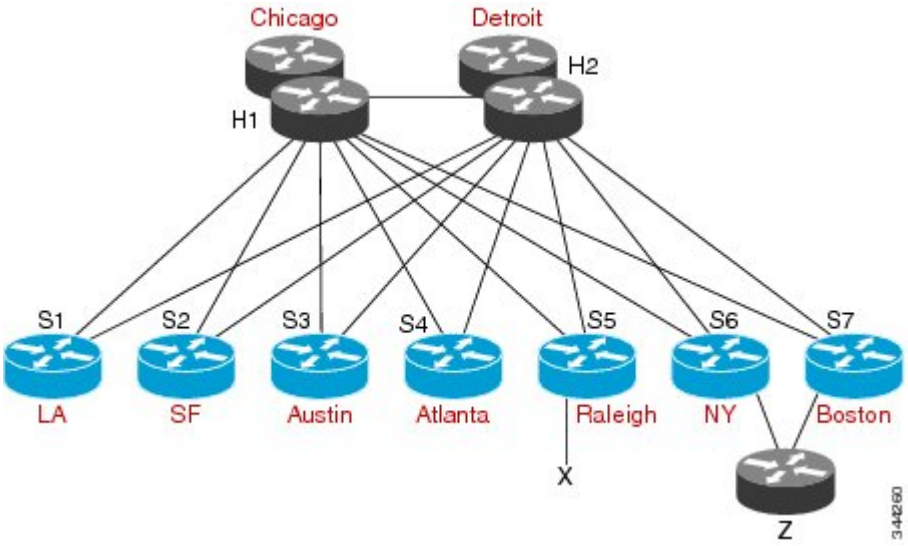


DMVPN Scenario

In Dynamic Multipoint Virtual Private Network (DMVPN) deployments, BGP is being used for scaling. In the figure below, Z is connected to both spokes S6 (NY) and S7 (Boston). The S7 links to the hubs have lower IGP costs than the S6 links to the hubs. There are physical links not shown that connect S5 to S6 and S6 to S7, with IGP costs lower than those to the hubs. Spokes S6 and S7 will send an update to both hubs H1 (Chicago) and H2 (Detroit). The RR hubs will then select the best path based on their lower IGP cost, which

might be S7. The spoke S5 (Raleigh) will receive two updates from the RRs for Z with S7 being the next hop, even though, in this scenario, it might be preferable to pick S6 (NY) as the next hop.

Figure 12: DMVPN Deployment



Benefits of BGP Additional Paths

BGP routers and route reflectors (RR) propagate only their best path over their sessions. The advertisement of a prefix replaces the previous announcement of that prefix (this in known as an implicit withdraw).

While this behavior may achieve better scaling, it can prevent path diversity, which tends to be poor or completely lost. The behavior in turn prevents efficient use of BGP multipath, prevents hitless planned maintenance, and can lead to multi-exit discriminator (MED) oscillations and suboptimal hot-potato routing. It also inhibits fast and local recovery upon nexthop failures, because the network has to wait for BGP control plane convergence to restore traffic.

The BGP Additional Paths feature is a BGP extension that allows the advertisement of multiple paths for the same prefix without the new paths implicitly replacing any previous paths. This behavior promotes path diversity and reduces MED oscillations.

BGP Additional Paths Functionality

The BGP Additional Paths feature is implemented by adding a path identifier to each path in the NLRI. The path identifier (ID) can be considered as something similar to a route distinguisher (RD) in VPNs, except that a path ID can apply to any address family. Path IDs are unique to a peering session and are generated for each network. The path identifier is used to prevent a route announcement from implicitly withdrawing the previous one. The Additional Paths feature allows the advertisement of more paths, in addition to the bestpath. The Additional Paths feature allows the advertisement of multiple paths for the same prefix, without the new paths implicitly replacing any previous paths.

The BGP Additional Paths feature requires the user to take three general steps:

1. Specify whether the device can send, receive, or send and receive additional paths. This is done at the address family level or the neighbor level, and is controlled by either the **bgp additional-paths** {**send** [**receive**] | **receive**} command or the **neighbor additional-paths** {**send** [**receive**] | **receive**} command, respectively. During session establishment, two BGP neighbors negotiate the Additional Path capabilities (whether they can send and/or receive) between them.

2. Select a set or sets of candidate paths for advertisement by specifying selection criteria (using the **bgp additional-paths select** command).

3. Advertise for a neighbor a set or sets of additional paths from the candidate paths marked (using the **neighbor advertise additional-paths** command).

To send or receive additional paths, the Additional Path capability must be negotiated. If it isn't negotiated, even if the selection criteria are such that more than the bestpath is marked and the neighbor is configured to advertise the marked paths, the selections would be useless because without the capability negotiated, only the bestpath can be sent.

Configuring BGP to send or receive additional paths triggers negotiation of additional path capability with the device's peers. Neighbors that have negotiated the capability will be grouped together in an update group (if other update group policies allow), and in a separate update group from those peers that have not negotiated the capability. Therefore, additional path capability causes the neighbor's update group membership to be recalculated.

Additional Path Selection

There are three path selection (path marking) policies, and they are not mutually exclusive. They are specified per address family, using the **bgp additional-paths select** command. They are:

- **best 2** or **best 3** (**best 2** means the bestpath and 2nd best path; the 2nd best path is the one computed by eliminating best-path from the best-computation algorithm. Similarly, **best 3** means the bestpath, 2nd best path, and 3rd best path; the 3rd best path is the one computed by eliminating bestpath and 2nd best path from the best-computation algorithm.)
- **group-best** (calculates the group-best for prefixes during bestpath calculation; described further below)
- **all** (all paths with unique next hops are eligible for selection)

Definition of the group-best Selection

The **group-best** keyword is part of the following commands:

- **advertise additional-paths**
- **bgp additional-paths select**
- **match additional-paths advertise-set**
- **neighbor advertise additional-paths**

The **group-best** is the set of paths that are the best paths from the paths of the same AS. For example, suppose there are three autonomous systems: AS 100, 200, and 300. Paths p101, p102, and p103 are from AS 100; p201, p202, and p203 are from AS200; and p301, p302, and p303 are from AS300. If we run the BGP bestpath algorithm on the paths from each AS, the algorithm will select one bestpath from each set of paths from that AS. Assuming p101 is the best from AS100, p201 is the best from AS200, and p301 is the best from AS300, then the **group-best** is the set of p101, p201, and p301.

Advertise a Subset of the Paths Selected

Take care when you select a set of paths but want to advertise a different set of paths. If the set of paths you want to advertise is not a subset of the selected paths, then you will not advertise the paths you want advertised.

The following example configures the additional paths selected to be the group-best and all selections. However, the paths configured to be advertised to the neighbor are the best 3 paths. Because the selection and advertise policy are not the same, the subsequent message is displayed. In these cases, only the bestpath is advertised.

```
Device(config)# router bgp 100
Device(config-router)# address-family ipv4
Device(config-router-af)# bgp additional-paths send receive
Device(config-router-af)# bgp additional-paths select group-best all
Device(config-router-af)# neighbor 192.168.2.2 advertise additional-paths best 3
 % BGP: AF level 'bgp additional-paths select' more restrictive than advertising policy.
This is a reminder that AF level additional-path select commands are needed.
```

How to Configure BGP Additional Paths

Configuring Additional Paths per Address Family

To select which paths are candidates to be additional paths, you can perform any combination of Steps 6, 7, and 8, as long as you perform at least one of those steps.

If you want to disable additional paths per neighbor, see the "Disabling Additional Paths per Neighbor" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4** [**unicast** | **multicast**]
5. **bgp additional-paths** {**send** [**receive**] | **receive**}
6. **bgp additional-paths select group-best**
7. **bgp additional-paths select best** *number*
8. **bgp additional-paths select all**
9. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name* } **advertise additional-paths** [**best** *number*] [**group-best**] [**all**]
10. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**address-family ipv4** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode. • The following address families are supported: IPv4 unicast, IPv4 multicast, IPv4 unicast + label, IPv6 unicast, IPv6 multicast, and IPv6 multicast + label.
Step 5	**bgp additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-af)# bgp additional-paths send receive`	Enables BGP additional paths to be sent only, received only, or sent and received, after negotiation with the neighbor is completed. • This example enables additional paths to be sent and received.
Step 6	**bgp additional-paths select group-best** **Example:** `Device(config-router-af)# bgp additional-paths select group-best`	(Optional) Calculates the group-best for prefixes during bestpath calculation.
Step 7	**bgp additional-paths select best** *number* **Example:** `Device(config-router-af)# bgp additional-paths select best 3`	(Optional) Calculates the specified number of best paths, including the advertisement of the bestpath. • The value of *number* can be 2 or 3.
Step 8	**bgp additional-paths select all** **Example:** `Device(config-router-af)# bgp additional-paths select all`	(Optional) Specifies that all paths with unique next hops are eligible for selection.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name* } **advertise additional-paths** [**best** *number*] [**group-best**] [**all**]	Specifies which selection methods control the additional paths that are advertised to the neighbor.

	Command or Action	Purpose
	Example: `Device(config-router-af)# neighbor 192.168.0.1 advertise additional-paths best 3 group-best all`	
Step 10	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits to privileged EXEC mode.

Configuring Additional Paths per Neighbor

To select which paths are candidates to be additional paths, you can perform any combination of Steps 6, 7, and 8, as long as you perform at least one of those steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4** [**unicast** | **multicast**]
5. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **additional-paths** {**send** [**receive**] | **receive**}
6. **bgp additional-paths select group-best**
7. **bgp additional-paths select best** *number*
8. **bgp additional-paths select all**
9. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **advertise additional-paths** [**best** *number*] [**group-best**] [**all**]
10. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**address-family ipv4** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode. • The following address families are supported: IPv4 unicast, IPv4 multicast, IPv4 unicast + label, IPv6 unicast, IPv6 multicast, and IPv6 multicast + label.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-af)# neighbor 192.168.1.2 additional-paths send receive`	Enables the neighbor to send or receive additional paths after negotiation is completed. • This example enables the neighbor to send and receive additional paths. • Note that this command overrides any send or receive capability that might have been configured at the address-family level.
Step 6	**bgp additional-paths select group-best** **Example:** `Device(config-router-af)# bgp additional-paths select group-best`	(Optional) Calculates the group-best for prefixes during bestpath calculation.
Step 7	**bgp additional-paths select best** *number* **Example:** `Device(config-router-af)# bgp additional-paths select best 3`	(Optional) Calculates the specified number of best paths, including the selection of the bestpath. • The value of *number* can be 2 or 3.
Step 8	**bgp additional-paths select all** **Example:** `Device(config-router-af)# bgp additional-paths select all`	(Optional) Specifies that all paths with unique next hops are eligible for selection.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **advertise additional-paths** [**best** *number*] [**group-best**] [**all**] **Example:** `Device(config-router-af)# neighbor 192.168.1.2 advertise additional-paths best 3 group-best all`	Specifies the selection methods that control which additional paths are advertised for the neighbor.
Step 10	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits to privileged EXEC mode.

Configuring Additional Paths Using a Peer Policy Template

In this configuration task example, the capability to send and receive additional paths and the selection criteria are configured for the address family, and then the template is configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4 unicast**
5. **bgp additional-paths** {**send** [**receive**] | **receive**}
6. **bgp additional-paths select** [**best** *number*] [**group-best**] [**all**]
7. **template peer-policy** *policy-template-name*
8. **additional-paths** {**send** [**receive**] | **receive**}
9. **advertise additional-paths** [**best** *number*] [**group-best**] [**all**]
10. **exit**
11. **address-family ipv4 unicast**
12. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
13. **neighbor** *ip-address* **inherit peer-policy** *policy-template-name*
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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 and creates a BGP routing process.
Step 4	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Configures the IPv4 address family.
Step 5	**bgp additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router)# bgp additional-paths send receive`	Enables BGP additional paths to be sent only, received only, or sent and received for the peers in the address family.
Step 6	**bgp additional-paths select** [**best** *number*] [**group-best**] [**all**] **Example:** `Device(config-router)# bgp additional-paths select best 3 group-best all`	Causes the system to calculate BGP additional paths that can be candidates for advertisement in addition to a bestpath.
Step 7	**template peer-policy** *policy-template-name* **Example:** `Device(config-router)# template peer-policy rr-client-pt1`	Enters policy-template configuration mode and creates a peer policy template.
Step 8	**additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-ptmp)# additional-paths send receive`	Enables BGP additional paths to be sent only, received only, or sent and received for the peers covered by the peer policy template.

	Command or Action	Purpose
Step 9	**advertise additional-paths** [**best** *number*] [**group-best**] [**all**] **Example:** `Device(config-router-ptmp)# advertise additional-paths best 3 group-best all`	Specifies the selection methods that control which additional paths are advertised for the peers covered by the peer policy template.
Step 10	**exit** **Example:** `Device(config-router-ptmp)# exit`	Exits policy-template configuration mode and returns to router configuration mode.
Step 11	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Configures the IPv4 address family.
Step 12	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 45000`	Adds an entry to the BGP neighbor table.
Step 13	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 inherit peer-policy rr-client-pt1`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration.
Step 14	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Filtering and Setting Actions for Additional Paths

You can optionally use a route map to filter the paths to be advertised by matching on the tags of additional paths that are candidates to be advertised. (These tags are the advertise-sets that are configured with the **bgp additional-paths select** command.) Paths that have the same path marking (tag) as the marking that is

configured in the **match additional-paths advertise-set** command match the route map entry (and are permitted or denied).

You can also optionally set one or more actions to take for those paths that pass the route map. This task happens to use the **set metric** command to illustrate using a route map with the **match additional-paths advertise-set** command. Of course, other **set** commands are available that are not shown in this task.

Why set a metric for paths marked with **all** (all paths with a unique next hop)? Suppose the neighbor 2001:DB8::1037 is receiving the same route from different neighbors. Routes received from the local device have a metric of 565 and routes from another device perhaps have a metric of 700. Routes with metric 565 will have precedence over the routes with metric 700.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
4. **match additional-paths advertise-set** [**best** *number*] [**best-range** *start-range end-range*] [**group-best**] [**all**]
5. **set metric** *metric-value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map additional_path1 permit 10`	Creates a route map.
Step 4	**match additional-paths advertise-set** [**best** *number*] [**best-range** *start-range end-range*] [**group-best**] [**all**] **Example:** `Device(config-route-map)# match additional-paths advertise-set best 3`	Matches on any path that is tagged with the specified path selection policy. • You must specify at least one selection method; you can specify more than one selection method in the command. • Specifying **best** *number* is incompatible with specifying **best-range**.

	Command or Action	Purpose
		• Specifying **best 1** will match only the bestpath. • Specifying **best-range 1 1** will match only the bestpath. • Only one **match additional-paths advertise-set** command is allowed per route map. A subsequent **match additional-paths advertise-set** command will overwrite the previous command.
Step 5	**set metric** *metric-value* **Example:** `Device(config-route-map)# set metric 500`	Sets the metric of the additional paths that pass the match criteria. • Note that other **set** commands can be used to take action on the paths that pass the route map. This example happens to use the **set metric** command.

What to Do Next

After creating the route map, you would reference the route map in the **neighbor route-map out** command. Thus, the route map is applied to paths being advertised (outgoing) to neighbors. Then you would use the **neighbor advertise additional-paths** command to advertise the additional paths. See the "Example: BGP Additional Paths" section to see the route map in context.

Displaying Additional Path Information

Perform either Step 2 or Step 3 in this task to see information about BGP additional paths.

SUMMARY STEPS

1. **enable**
2. **show ip bgp neighbors** [*ip-address*]
3. **show ip bgp** [*network*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**show ip bgp neighbors** [*ip-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.1`	Displays the capabilities of the neighbor to send and receive additional paths.
Step 3	**show ip bgp** [*network*] **Example:** `Device# show ip bgp 192.168.0.0`	Displays the additional path selections and path ID for the network.

Disabling Additional Paths per Neighbor

If you had configured the sending or receiving of additional paths on a per neighbor basis (with the **neighbor additional-paths** command), and you wanted to disable that functionality, you would use the **no neighbor additional-paths** command.

However, if you had configured the sending or receiving of additional paths for an address family (with the **bgp additional-paths** command), and you wanted to disable that functionality for a neighbor, you would use the **neighbor additional-paths disable** command. Disabling additional paths also works if the functionality was inherited from a template.

Perform this task to disable additional path capability for a neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**]
5. **bgp additional-paths** {**send** [**receive**] | **receive**}
6. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **additional-paths disable**
7. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 unicast`	Enters address family configuration mode.
Step 5	**bgp additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-af)# bgp additional-paths send receive`	Enables BGP additional paths to be sent or received for the neighbors in the address family.
Step 6	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **additional-paths disable** **Example:** `Device(config-router-af)# neighbor 2001:DB8::1 additional-paths disable`	Disables BGP additional paths from being sent to or received from the specified neighbor. • The additional path functionality is still enabled for the rest of the neighbors in the address family.
Step 7	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits to privileged EXEC mode.

Configuration Examples for BGP Additional Paths

Example: BGP Additional Path Send and Receive Capabilities

In this example, R1's address is 192.168.1.1; its neighbor is R2, which has address 192.168.1.2. Updates are sent from R2 to R1 with additional-paths (all paths advertised). Updates are sent from R1 to R2 with only the classic BGP bestpath advertised because R2 is only able to send additional paths, not receive additional paths.

R1

```
router bgp 1
 address-family ipv4 unicast
 bgp additional-paths select all
 neighbor 192.168.1.2 additional-paths send receive
 neighbor 192.168.1.2 advertise additional-paths all
```

R2

```
router bgp 2
 address-family ipv4 unicast
 bgp additional-paths select all
 neighbor 192.168.1.1 additional-paths send
 neighbor 192.168.1.1 advertise additional-paths all
```

Example: BGP Additional Paths

In the following example, for every address family, there are one or more eBGP neighbors not shown in the configuration that are sending routes to the local device. The eBGP routes learned from those neighbors are advertised toward the neighbors shown in the configuration below and the path attributes are changed. The example configures that:

- The route map called add_path1 specifies that all the paths are advertised toward neighbor 192.168.101.15, but any path that is marked with **best 2** will have its metric set to 780 before being sent toward that neighbor.
- The route map called add_path2 specifies that any path that is marked with **best 3** will have its metric set to 640 and will be advertised toward neighbor 192.168.25.
- The route map called add_path3 specifies that any path that is marked with **group-best** will have its metric set to 825 and will be advertised toward neighbor 2001:DB8::1045.
- In the IPv6 multicast address family, all paths are candidates to be advertised and will be advertised toward neighbor 2001:DB8::1037.

```
router bgp 1
 neighbor 192.168.101.15 remote-as 1
 neighbor 192.168.101.25 remote-as 1
 neighbor 2001:DB8::1045 remote-as 1
 neighbor 2001:DB8::1037 remote-as 1
!
 address-family ipv4 unicast
  bgp additional-paths send receive
  bgp additional-paths select all best 3 group-best
  neighbor 192.168.101.15 activate
  neighbor 192.168.101.15 route-map add_path1 out
  neighbor 192.168.101.15 advertise additional-paths best 2
 exit-address-family
!
address-family ipv4 multicast
```

```
  bgp additional-paths send receive
  bgp additional-paths select all best 3 group-best
  neighbor 192.168.101.25 activate
  neighbor 192.168.101.25 route-map add_path2 out
  neighbor 192.168.101.25 advertise additional-paths best 3
 exit-address-family
!
 address-family ipv6 unicast
  bgp additional-paths send receive
  bgp additional-paths select group-best
  neighbor 2001:DB8::1045 activate
  neighbor 2001:DB8::1045 route-map add_path3 out
  neighbor 2001:DB8::1045 advertise additional-paths all group-best
 exit-address-family
!
 address-family ipv6 multicast
  bgp additional-paths send receive
  bgp additional-paths select all
  neighbor 2001:DB8::1037 activate
  neighbor 2001:DB8::1037 route-map add_path4 out
  neighbor 2001:DB8::1037 advertise additional-paths all
 exit-address-family
!
route-map add_path1 permit 10
 match additional-paths advertise-set best 2
 set metric 780
route-map add_path1 permit 20
!
route-map add_path2 permit 10
 match additional-paths advertise-set best 3
 set metric 640
!
route-map add_path3 permit 10
 match additional-paths advertise-set group-best
 set metric 825
!
```

Example: Neighbor Capabilities Override Address Family Capabilities

In the following example, the receive-only capability of the neighbor overrides the send and receive capability of the address family:

```
router bgp 65000
 address-family ipv6 multicast
 bgp additional-paths send receive
 bgp additional-paths select group-best
 neighbor 2001:DB8::1037 activate
 neighbor 2001:DB8::1037 additional-paths receive
 neighbor 2001:DB8::1037 advertise additional-paths group-best
!
```

Example: BGP Additional Paths Using a Peer Policy Template

```
router bgp 45000
address-family ipv4 unicast
 bgp additional-paths send receive
 bgp additional-paths select all group-best best 3
 template peer-policy rr-client-pt1
  additional-paths send receive
  advertise additional-paths group-best best 3
  exit
 address-family ipv4 unicast
  neighbor 192.168.1.1 remote-as 45000
  neighbor 192.168.1.1 inherit peer-policy rr-client-pt1
```

```
end
```

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 3107	*Carrying Label Information in BGP-4*
RFC 4271	*A Border Gateway Protocl (BGP-4)*
RFC 4760	*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

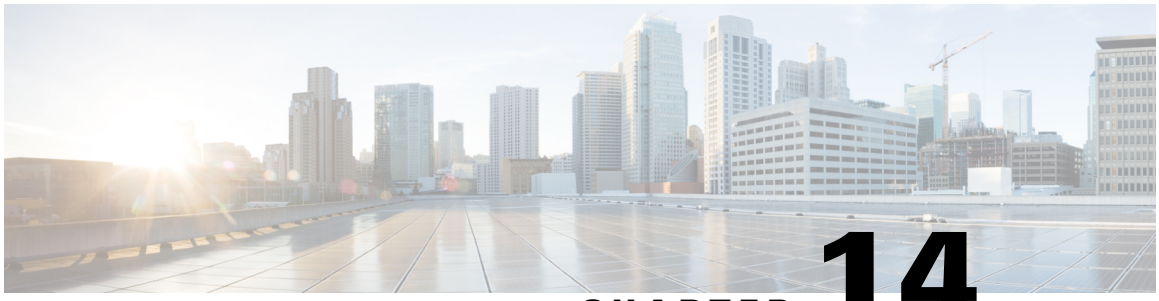
Feature Information for BGP Additional Paths

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 16: Feature Information for BGP Additional Paths

Feature Name	Releases	Feature Information
BGP Additional Paths	15.2(1)E	The BGP Additional Paths feature allows the advertisement of multiple paths for the same prefix without the new paths implicitly replacing any previous paths. The following commands were introduced: • **additional-paths** • **advertise additional-paths** • **bgp additional-paths** • **bgp additional-paths select** • **match additional-paths advertise-set** • **neighbor additional-paths** • **neighbor advertise additional-paths** The following commands were modified: • **show ip bgp** • **show ip bgp neighbors**



CHAPTER 14

BGP Attribute Filter and Enhanced Attribute Error Handling

The BGP Attribute Filter feature allows you to "treat-as-withdraw" updates that contain specific path attributes. The prefixes contained in the update are removed from the routing table. The feature also allows you to remove specific path attributes from incoming updates. Both behaviors provide an increased measure of security. The BGP Enhanced Attribute Error Handling feature prevents peer sessions from flapping due to errors from any malformed update, thereby saving resources.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About BGP Attribute Filtering

BGP Attribute Filter and Enhanced Attribute Error Handling

The BGP Attribute Filter feature provides two ways to achieve an increased measure of security:

BGP Attribute Filter and Enhanced Attribute Error Handling

- The feature allows you to treat-as-withdraw an Update coming from a specified neighbor if the Update contains a specified attribute type. When an Update is treat-as-withdraw, the prefixes in the Update are removed from the BGP routing table (if they existed in the routing table).
- The feature also allows you to drop specified path attributes from an Update, and then the system processes the rest of the Update as usual.

The BGP Enhanced Attribute Error Handling feature prevents peer sessions from flapping due to a malformed Update. The malformed Update is treat-as-withdraw and does not cause the BGP session to be reset. This feature is enabled by default, but can be disabled.

The features are implemented in the following order:

1. Received Updates that contain user-specified path attributes are treat-as-withdraw (as long as the NLRI can be parsed successfully). If there is an existing prefix in the BGP routing table, it will be removed. The **neighbor path-attribute treat-as-withdraw** command configures this feature.
2. User-specified path attributes are discarded from received Updates, and the rest of the Update is processed normally. The **neighbor path-attribute discard** command configures this feature.
3. Received Updates that are malformed are treat-as-withdraw. This feature is enabled by default; it can be disabled by configuring the **no bgp enhanced-error** command.

Details About Specifying Attributes as Treat-as-Withdraw

Attribute types 1, 2, 3, 4, 8, 14, 15, and 16 cannot be configured for path attribute treat-as-withdraw.

Attribute type 5 (localpref), type 9 (Originator,) and type 10 (Cluster-id) can be configured for treat-as-withdraw for eBGP neighbors only.

Configuring path attributes to be treated as withdrawn will trigger an inbound Route Refresh to ensure that the routing table is up to date.

Details About Specifying Attributes as Discard

Attribute types 1, 2, 3, 4, 8, 14, 15, and 16 cannot be configured for path attribute discard.

Attribute type 5 (localpref), type 9 (Originator), and type 10 (Cluster-id) can be configured for discard for eBGP neighbors only.

Configuring path attributes to be discarded will trigger an inbound Route Refresh to ensure that the routing table is up to date.

Details About Enhanced Attribute Error Handling

If a malformed Update is received, it is treat-as-withdraw to prevent peer sessions from flapping due to the processing of BGP path attributes. This feature applies to eBGP and iBGP peers. This feature is enabled by default; it can be disabled.

If the BGP Enhanced Attribute Error Handling feature is enabled or disabled, BGP places the MP_REACH attribute (attribute 14) at the beginning of an attribute list while formatting an update. Enhanced attribute error handling functions more easily when the MP_REACH attribute is at the beginning of the attribute list.

How to Filter BGP Path Attributes

Treat-as-Withdraw BGP Updates Containing a Specified Path Attribute

Note Performing this task will trigger an inbound Route Refresh to ensure that the routing table is up to date.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*} **path-attribute treat-as-withdraw** {*attribute-value* | **range** *start-value end-value*} **in**
5. Repeat Step 4 to configure other attributes not in a range or to configure a different neighbor.
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process and enters router configuration mode.
Step 4	**neighbor** {*ip-address* | *ipv6-address*} **path-attribute treat-as-withdraw** {*attribute-value* | **range** *start-value end-value*} **in** **Example:** `Device(config-router)# neighbor 2001:DB8:1::1 path-attribute treat-as-withdraw 100 in`	Treat-as-withdraw any incoming Update messages that contain the specified path attribute or range of path attributes. • Any prefixes in an Update that is treat-as-withdraw are removed from the BGP routing table.

	Command or Action	Purpose
		• The specific attribute value and the range of attribute values are independent of each other.
Step 5	Repeat Step 4 to configure other attributes not in a range or to configure a different neighbor.	
Step 6	**end** **Example:** `Device(config-router)# end`	Exits to privileged EXEC mode.

Discarding Specific Path Attributes from an Update Message

Note Performing this task will trigger an inbound Route Refresh to ensure that the routing table is up to date.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*} **path-attribute discard** {*attribute-value* | **range** *start-value end-value*} **in**
5. Repeat Step 4 to configure other attributes not in a range or to configure a different neighbor.
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 6500`	Configures a BGP routing process and enters router configuration mode.
Step 4	**neighbor** {*ip-address* | *ipv6-address*} **path-attribute discard** {*attribute-value* | **range** *start-value end-value*} **in** **Example:** `Device(config-router)# neighbor 2001:DB8:1::1 path-attribute discard 128 in`	Drops specified path attributes from Update messages from the specified neighbor.
Step 5	Repeat Step 4 to configure other attributes not in a range or to configure a different neighbor. **Example:**	
Step 6	**end** **Example:** `Device(config-router)# end`	Exits to privileged EXEC mode.

Displaying Withdrawn or Discarded Path Attributes

Perform any of these steps in any order to display information about treat-as-withdraw, discarded, or unknown path attributes. You can use the **show ip bgp** command with any address family that BGP supports, such as **show ip bgp ipv4 multicast**, **show ip bgp ipv6 unicast**, etc.

SUMMARY STEPS

1. **enable**
2. **show ip bgp neighbor** [*ip-address* | *ipv6-address*]
3. **show ip bgp path-attribute unknown**
4. **show ip bgp path-attribute discard**
5. **show ip bgp vpnv4 all** *prefix*
6. **show ip bgp neighbors** *prefix*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp neighbor** [*ip-address* | *ipv6-address*] **Example:** `Device# show ip bgp neighbor 2001:DB8:1::1`	(Optional) Displays the configured discard and treat-as-withdraw attribute values for the neighbor, counts of Updates with such attributes discarded or treat-as-withdraw, and the count of malformed treat-as-withdraw Updates.
Step 3	**show ip bgp path-attribute unknown** **Example:** `Device# show ip bgp path-attribute unknown`	(Optional) Displays all prefixes that have an unknown attribute.
Step 4	**show ip bgp path-attribute discard** **Example:** `Device# show ip bgp path-attribute discard`	(Optional) Displays all prefixes for which an attribute has been discarded.
Step 5	**show ip bgp vpnv4 all** *prefix* **Example:** `Device# show ip bgp vpnv4 all 192.168.1.0`	(Optional) Displays the unknown attributes and discarded attributes associated with a prefix.
Step 6	**show ip bgp neighbors** *prefix* **Example:** `Device# show ip bgp neighbors 192.168.1.0`	(Optional) Displays the configured discard and treat-as-withdraw attributes associated with a prefix.

Configuration Examples for BGP Attribute Filter

Examples: Withdraw Updates Based on Path Attribute

The following example shows how to configure the device to treat-as-withdraw any Update messages from the specified neighbor that contain the unwanted path attribute 100 or 128:

```
router bgp 65600
 neighbor 2001:DB8:1::2 path-attribute treat-as-withdraw 100 in
 neighbor 2001:DB8:1::2 path-attribute treat-as-withdraw 128 in
```

The following example shows how to configure the device to treat-as-withdraw any Update messages from the specified neighbor that contain the unwanted path attributes in the range from 21 to 255:

```
router bgp 65600
 neighbor 2001:DB8:1::2 path-attribute treat-as-withdraw 21 255 in
```

Examples: Discard Path Attributes from Updates

The following example shows how to configure the device to discard path attributes 100 and 128 from incoming Update messages from the specified neighbor. The rest of the Update message will be processed as usual.

```
router bgp 65600
 neighbor 2001:DB8:1::1 path-attribute discard 100 in
 neighbor 2001:DB8:1::1 path-attribute discard 128 in
```

The following example shows how to configure the device to discard path attributes in the range from 17 to 255 from incoming Update messages from the specified neighbor. The rest of the Update message will be processed as usual.

```
router bgp 65600
 neighbor 2001:DB8:1::1 path-attribute discard 17 255 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
draft-ietf-idr-error-handling	Revised Error Handling for BGP Updates from External Neighbors

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

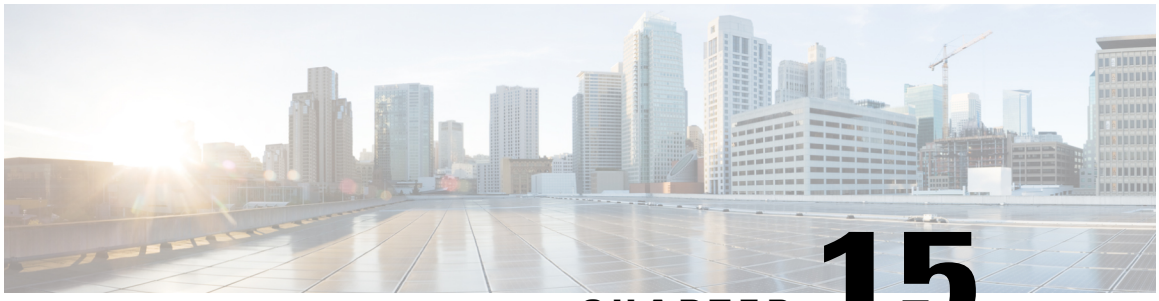
Feature Information for BGP Attribute Filter and Enhanced Attribute Error Handling

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 17: Feature Information for BGP Attribute Filter and Enhanced Attribute Error Handling

Feature Name	Releases	Feature Information
BGP Attribute Filter and Enhanced Attribute Error Handling	15.2(1)E	The BGP Attribute Filter allows you to "treat-as-withdraw" updates that contain specific path attributes. The prefixes contained in the update are removed from the routing table. The feature also allows you to remove specific path attributes from incoming updates. Both behaviors provide an increased measure of security. The BGP Enhanced Attribute Error Handling feature prevents peer sessions from flapping due to errors from any malformed update, thereby saving resources. The following commands were introduced: **bgp enhanced-error**, **neighbor path-attribute discard**, **neighbor path-attribute treat-as-withdraw**, **show ip bgp path-attribute discard**, and **show ip bgp path-attribute unknown**. The following commands were modified: **show ip bgp**, **show ip bgp neighbor**, and **show ip bgp vpnv4 all**.



CHAPTER 15

BGP Support for IP Prefix Export from a VRF Table into the Global Table

This feature allows a network administrator to export IP prefixes from a VRF table into the global routing table.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IP Prefix Export from a VRF Table into the Global Table

Benefits of IP Prefix Export from a VRF Table into the Global Table

- You can manage some network resources inside a VRF by using a network management node residing in the global table.
- You own some internet public IP address space, but prefer to have a VRF to manage those IP addresses.

How IP Prefix Export from a VRF Table into the Global Table Works

MPLS-VPN using Multiprotocol BGP (MP-BGP) provides a very flexible but secured VPN provisioning mechanism for service providers and customers. However, some customers prefer to relax the boundary so that some specific prefixes can be reachable in a VRF as well as in the global routing table.

Prior to the BGP Support for IP Prefix Export from a VRF Table into Global Table feature, BGP already supported the global-to-VRF import of prefixes. See the "*BGP Support for IP Prefix Import from Global Table into a VRF Table*" module for complete documentation of that feature. Together, the import feature and export feature provide L3VPN dynamic route leaking.

The BGP Support for IP Prefix Export from a VRF Table into the Global Table feature provides the reverse mechanism of the import feature referenced above; it supports the export of prefixes from a VRF table to the global routing table. It is achieved with an **export {ipv4 | ipv6} {unicast | multicast} map** command, which specifies a route map to control the prefixes that are exported from a VRF table to the global routing table.

Caution The IP Prefix Export from a VRF Table into Global Table feature leaks VRF routes into the global BGP routing table; those routes will be installed into the IPv4 or IPv6 routing table. Use extreme caution to design the network so that such leaking does not affect the normal Internet routing.

Export actions are triggered when a new routing update is received or when routes are withdrawn. During the initial BGP update period, the export action is postponed to allow BGP to converge more quickly. Once BGP converges, incremental BGP updates are evaluated immediately and qualified prefixes are exported as they are received.

Each VRF can export to only one of the global topologies in IPv4 (unicast or multicast) and can export to only one of the global topologies in IPv6 (unicast or multicast).

There is no limit to the number of VRFs per router that can be configured to export IPv4 or IPv6 prefixes to the global routing table.

By default, the software limits the number of prefixes that can be exported per VRF to 1000 prefixes. You can change that limit to a number in the range from 1 to 2,147,483,647 prefixes for each VRF. We recommend that you use caution if you increase the prefix limit above 1000. Configuring the device to export too many prefixes can interrupt normal router operation.

The following **match** and **set** commands are supported in this feature:

- **match as-path**

- **match community** [**exact-match**]
- **match extcommunity**
- **match ip address** [**prefix-list**]
- **match ip next-hop**
- **match ip route-source**
- **match ipv6 address** [**prefix-list**]
- **match ipv6 route-source**
- **match ipv6 next-hop**
- **match policy-list**
- **match route-type**
- **set as-path prepend** [**last-as**]
- **set community additive**
- **set extcommunity** [**cost** | **rt**]
- **set extcomm-list delete**
- **set ip next-hop**
- **set ipv6 next-hop**
- **set local-preference**
- **set metric**
- **set origin**
- **set weight**

Note The **set ip vrf next-hop** and **set ipv6 vrf next-hop** commands are not supported in this feature.

How to Export IP Prefixes from a VRF Table into the Global Table

Creating the VRF and the Export Route Map for an Address Family

The IP prefixes that are defined for export are processed through a match clause in a route map. IP prefixes that pass through the route map are exported into the global routing table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf-name*
4. **rd** *route-distinguisher*
5. **address-family** {**ipv4** | **ipv6**}
6. **export** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*prefix-limit*] **map** *map-name*
7. **route-target import** *route-target-ext-community*
8. **route-target export** *route-target-ext-community*
9. **exit**
10. **exit**
11. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
12. **match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]}
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vpn1`	Creates a VRF routing table and specifies the VRF name (or tag).
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:100`	Creates routing and forwarding tables for the VRF instance. • There are two formats for configuring the argument. It can be configured in the *as-number*:*network number* (*ASN*:*nn*) format, as shown in the example, or it can be configured in the *IP address*:*network number* format (*IP-address*:*nn*).

	Command or Action	Purpose
Step 5	**address-family** {**ipv4** | **ipv6**} **Example:** `Device(config-vrf)# address-family ipv4`	Configures the IPv4 or IPv6 address family.
Step 6	**export** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*prefix-limit*] **map** *map-name* **Example:** `Device(config-vrf-af)# export ipv4 unicast 500 map UNICAST`	Exports IPv4 or IPv6 prefixes from the VRF table to the global routing table, filtered by the specified route map. • Specify **ipv4** or **ipv6**, which you specified in Step 5. This example exports IPv4 unicast prefixes. • Based on this example, no more than 500 prefixes will be exported. • The prefixes exported are those that pass the route map.
Step 7	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf-af)# route-target import 100:100`	Creates a route-target extended community for a VRF instance. • For information about route-target import or export, see the *MPLS: Layer 3 VPNs Configuration Guide*.
Step 8	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf-af)# route-target export 100:100`	Creates a route-target extended community for a VRF instance.
Step 9	**exit** **Example:** `Device(config-vrf-af)# exit`	Exits address family configuration mode and enters global configuration mode.
Step 10	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 11	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map UNICAST permit 10`	Enables policy routing. • The example creates a route map named UNICAST.

	Command or Action	Purpose
Step 12	**match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]} **Example:** `Device(config-route-map)# match ip address 50`	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, and performs policy routing on matched packets. • Both IP access lists and IP prefix lists are supported. • The example configures the route map to use standard access list 50 to define match criteria. • Define the access list (not shown in this task); for example, access-list 50 permit 192.168.1.0 255.255.255.0.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and returns to privileged EXEC mode.

Creating the VRF and the Export Route Map for a VRF (IPv4 only)

The IP prefixes that are defined for export are processed through a match clause in a route map. IP prefixes that pass through the route map are exported into the global routing table.

Note

- Only IPv4 unicast and multicast prefixes can be exported from a VRF table to the global routing table under the **ip vrf** command, as shown in this task. To export IPv6 prefixes, you must do so under the IPv6 address family; see the section "Creating the VRF and the Export Route Map Per Address Family."
- IPv4 prefixes exported into the global routing table using this feature cannot be exported into a VPNv4 VRF.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*
5. **export ipv4** {**unicast** | **multicast**} [*prefix-limit*] **map** *map-tag*
6. **route-target import** *route-target-ext-community*
7. **route-target export** *route-target-ext-community*
8. **exit**
9. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
10. **match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]}
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf GREEN`	Creates a VRF routing table and specifies the VRF name (or tag). • The **ip vrf** *vrf-name* command creates a VRF routing table and a CEF table, and both are named using the *vrf-name* argument. Associated with these tables is the default route distinguisher value.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:10`	Creates routing and forwarding tables for the VRF instance. • There are two formats for configuring the argument. It can be configured in the *as-number***:***network number* (*ASN***:***nn*) format, as shown in the example, or it can be configured in the *IP-address***:***network number* format (*IP-address***:***nn*).
Step 5	**export ipv4** {**unicast**	**multicast**} [*prefix-limit*] **map** *map-tag*	Exports IPv4 prefixes from the VRF table to the global routing table, filtered by the specified route map.

	Command or Action	Purpose
	Example: `Device(config-vrf)# export ipv4 unicast 500 map UNICAST`	• Unicast or multicast prefixes are specified. • By default, up to 1000 prefixes can be exported. The *prefix-limit* argument is used to specify a limit from 1 to 2,147,483,647 prefixes. • The example creates an export map that will export up to 500 unicast prefixes that pass through the route map named UNICAST.
Step 6	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 100:100`	Creates a route-target extended community for a VRF instance. • For information about route-target import or export, see the *MPLS: Layer 3 VPNs Configuration Guide*.
Step 7	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 100:100`	Creates a route-target extended community for a VRF instance.
Step 8	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 9	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map UNICAST permit 10`	Defines the conditions for redistributing routes from one routing protocol into another, or enables policy routing. • The route map name must match the route map specified in Step 5. • The example creates a route map named UNICAST.
Step 10	**match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]} **Example:** `Device(config-route-map)# match ip address 50`	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, and performs policy routing on matched packets. • Both IP access lists and IP prefix lists are supported. • The example configures the route map to use standard access list 50 to define match criteria.

	Command or Action	Purpose
Step 11	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and returns to privileged EXEC mode.

Displaying Information About IP Prefix Export from a VRF into the Global Table

Perform any of the steps in this task to see information about the prefixes exported from a VRF table into the global table.

SUMMARY STEPS

1. **enable**
2. **show ip bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*prefix*]
3. **debug ip bgp import event**
4. **debug ip bgp import update**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp** {**ipv4**	**ipv6**} {**unicast**	**multicast**} [*prefix*] **Example:** `Device# show ip bgp ipv4 unicast 192.168.1.1`	Displays information about the imported path from a VRF to the global table.
Step 3	**debug ip bgp import event** **Example:** `Device# debug ip bgp import event`	Displays messages related to IPv4 prefix import events.
Step 4	**debug ip bgp import update** **Example:** `Device# debug ip bgp import update`	Displays messages related to IPv4 prefix import updates.

Configuration Examples for IP Prefix Export from a VRF Table into the Global Table

Example: Exporting IP Prefixes from a VRF Table into the Global Table Using IPv6 Address Family

```
vrf definition X
  rd 100:100
   address-family ipv6
    export ipv6 unicast map OnlyNet2000
    route-target import 100:100
    route-target export 100:100
!
 ipv6 prefix-list net2000 permit 2000::/16
!
route-map OnlyNet2000 permit 10
 match ipv6 address prefix-list net2000
```

Example: Exporting IP Prefixes from a VRF Table into the Global Table Using IPv4 Address Family

```
vrf definition X
  rd 100:100
   address-family ipv4
    export ipv4 unicast map OnlyNet200
    route-target import 100:100
    route-target export 100:100
!
 ip prefix-list net200 permit 200.0.0.0/8
!
route-map OnlyNet200 permit 10
 match ip address prefix-list net200
```

Example: Exporting IP Prefixes from a VRF Table into the Global Table Using IP VRF (IPv4 Only)

```
 ip vrf vrfname
  rd 100:100
  export ipv4 unicast map OnlyNet200
  route-target import 100:100
  route-target export 100:100
!
 ip prefix-list net200 permit 200.0.0.0/8
!
route-map OnlyNet200 permit 10
 match ip address prefix-list net200
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS BGP Command Reference
Use of route-target import and export	*MPLS: Layer 3 VPNs 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

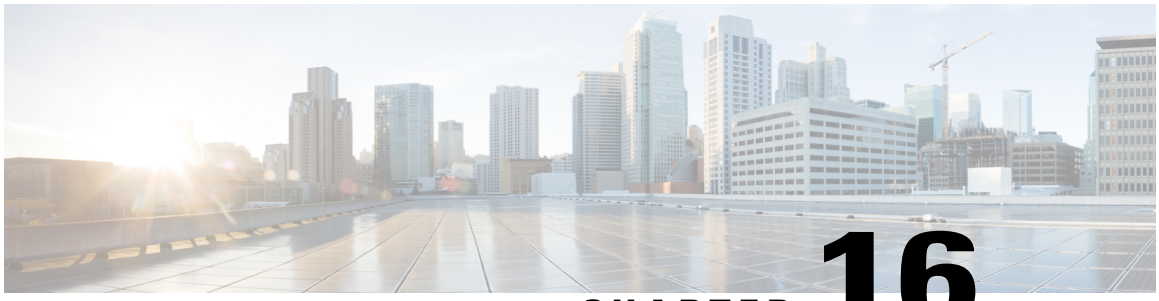
Feature Information for IP Prefix Export from a VRF Table into the Global Table

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 18: Feature Information for BGP Support for IP Prefix Export from a VRF Table into the Global Table

Feature Name	Releases	Feature Information
BGP Support for IP Prefix Export from a VRF Table into the Global Table	15.2(1)E	This feature allows a network administrator to export IP prefixes from a VRF routing table into the global routing table. The following command was introduced: **export map (VRF table to global table)**. The following commands were modified: **debug ip bgp import** and **show ip bgp**.



CHAPTER 16

Cisco-BGP-MIBv2

The Cisco-BGP-MIBv2 feature supports CISCO-BGP-MIB, which allows MIBv2 traps to be generated. CISCO-BGP-MIB supports IPv4, IPv6, VPNv4, and VPNv6 address families. Its MIB objects and SNMP notifications can keep track of prefix counters and capabilities related to BGP sessions.

Finding Feature Information

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

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Cisco-BGP-MIBv2

Benefits of Cisco-BGP-MIBv2

The benefits of CISCO-BGP-MIB are as follows:

- MIB objects and SNMP notifications can keep track of prefix counters and capabilities related to BGP sessions.
- The following address families are supported: IPv4, IPv6, VPNv4, VPNv6.

- New traps are added to CISCO-BGP4-MIB that allow data to be returned for both IPv4 and IPv6 peers.

Cisco-BGP-MIBv2 Read-Only Objects

Cisco BGP Peer Table—cbgpPeer2Table

This table is an augmentation of the former Cisco BGP Peer Table, with flexible indexing to support both IPv4 and IPv6. This table contains one entry per BGP peer about the connections with BGP peers. MIB objects in this table are:

- cbgpPeer2Type
- cbgpPeer2RemoteAddr
- cbgpPeer2State
- cbgpPeer2AdminStatus
- cbgpPeer2NegotiatedVersion
- cbgpPeer2LocalAddr
- cbgpPeer2LocalPort
- cbgpPeer2LocalAs
- cbgpPeer2LocalIdentifier
- cbgpPeer2RemotePort
- cbgpPeer2RemoteAS
- cbgpPeer2RemoteIdentifier
- cbgpPeer2InUpdates
- cbgpPeer2OutUpdates
- cbgpPeer2InTotalMessages
- cbgpPeer2OutTotalMessages
- cbgpPeer2LastError
- cbgpPeer2FsmEstablishedTransitions
- cbgpPeer2FsmEstablishedTime
- cbgpPeer2ConnectRetryInterval
- cbgpPeer2HoldTime
- cbgpPeer2KeepAlive
- cbgpPeer2HoldTimeConfigured
- cbgpPeer2KeepAliveConfigured
- cbgpPeer2MinAsOriginationInterval
- cbgpPeer2MinRouteAdvertisementInterval
- cbgpPeer2InUpdateElapsedTime

- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

BGP Session Capability Table—cbgpPeer2CapsTable

This table is an augmentation of the former Cisco BGP Session Capability Table, with flexible indexing to support both IPv4 and IPv6. BGP sessions can have multiple capabilities, which are sent to the peer in BGP OPEN messages. BGP_CAP_TYPE_AS4 is the only new capability support added from the previous version of cbgpPeerCapsTable. This table contains an entry corresponding to a capability code and an index. The MIB object in this table is:

- cbgpPeer2CapCode

Peer Address Family Table—cbgpPeer2AddrFamilyTable

This table is an augmentation of the former Cisco Peer Address Family Table, with flexible indexing to support both IPv4 and IPv6. This table contains supported address families. Only the following address families are supported: IPv4, VPNv4, IPv6, and VPNv6. Output is similar to the original MIB Object Identifier (OID): cbgpPeerAddrFamilyTable. The MIB object in this table is:

- cbgpPeer2AddrFamilyName

Prefix CounterTable—cbgpPeer2AddrFamilyPrefixTable

This table is an augmentation of the former Cisco Prefix Counter Table, with flexible indexing to support both IPv4 and IPv6. This table contains per peer route prefix counters for all of the supported address families. Only the following address families are supported: IPv4, VPNv4, IPv6, and VPNv6. Output is similar to the original MIB OID: cbgpPeerAddrFamilyPrefixTable. The MIB objects in this table are:

- cbgpPeer2AcceptedPrefixes
- cbgpPeer2DeniedPrefixes
- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixThreshold
- cbgpPeer2PrefixClearThreshold
- cbgpPeer2AdvertisedPrefixes
- cbgpPeer2SuppressedPrefixes
- cbgpPeer2WithdrawnPrefixes

cbgpNotifsEnable

This global object contains information about which traps are enabled. When the **snmp-server enable traps bgp cbgp2** command is configured, the following traps are sent:

cbgpPeer2FsmStateChange

This notification is an augmentation of the former FSM State Change Notification, with support for IPv6. This notification contains the following MIB objects:

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt

- cbgpPeer2PrevState

cbgpPeer2BackwardTransition

This notification is an augmentation of the former Backward Transition Notification, with support for IPv6. This is sent when BGP FSM moves from a higher numbered state to a lower numbered state. This notification contains the following MIB objects:

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2PrefixThresholdExceeded

This notification is an augmentation of the former Prefix Threshold Exceeded Notification, with support for IPv6. This is sent when the prefix count for an address family on a BGP session exceeds the configured threshold value. This notification contains the following MIB objects:

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixThreshold

cbgpPeer2PrefixThresholdClear

This notification is an augmentation of the former Prefix Threshold Clear Notification, with support for IPv6. It is sent when the prefix count drops below the clear trap limit for an address family on a BGP session after the generation of cbgpPeer2PrefixThresholdExceeded notification. This notification contains the following MIB objects:

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixClearThreshold

cbgpPeer2BackwardTransNotification

This notification is generated when the BGP FSM moves from a higher numbered state to a lower numbered state. It contains the following MIB objects:

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

Cisco-BGP-MIBv2 Trap Objects

cbgpPeer2EstablishedNotification

This notification is generated when the BGP FSM enters the established state. The MIB objects are:

- cbgpPeer2LastError
- cbgpPeer2State

cbgpPeer2BackwardTransNotification

This notification is generated when the BGP FSM moves from a higher numbered state to a lower numbered state. The MIB objects are:

- cbgpPeer2LastError
- cbgpPeer2State

cbgpPeer2FsmStateChange

This notification is generated for every BGP FSM state change. The MIB objects are:

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2BackwardTransition

This notification is generated when the BGP FSM moves from a higher numbered state to a lower numbered state. The MIB objects are:

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2PrefixThresholdExceeded

This notification is generated when the prefix count exceeds the configured warning threshold on a session for an address family. The MIB objects are:

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixThreshold

cbgpPeer2PrefixThresholdClear

This notification is generated when the prefix count drops below the configured clear threshold on a session for an address family once cbgpPeer2PrefixThresholdExceeded is generated. This notification is not generated if the peer session goes down after the generation of cbgpPrefixThresholdExceeded. The MIB objects are:

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixClearThreshold

How to Configure Cisco-BGP-MIBv2

Enabling Cisco-BGP-MIBv2 Traps

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server enable traps bgp cbgp2** [[**state-changes** [**all**] [**backward-trans**] [**limited**]] | [**threshold prefix**]]
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server enable traps bgp cbgp2** [[**state-changes** [**all**] [**backward-trans**] [**limited**]]	[**threshold prefix**]] **Example:** `Device(config)# snmp-server enable traps bgp cbgp2`	Enables generation of Cisco-BGP-MIBv2 traps. • The **bgp** keyword enables generation of original MIB traps. • The **bgp cbgp2** keywords enable generation of MIBv2 traps. • The original MIB and MIBv2 can be enabled or disabled individually. If both the original MIB and MIBv2 are enabled, both traps are generated. If only one of the MIB versions is enabled, only that version of traps is generated.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.

Configuration Examples for Cisco-BGP-MIBv2

Example: Enabling Cisco-BGP-MIBv2

```
Device(config)# snmp-server enable traps bgp cbgp2
```

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
CISCO-BGP4-MIB support	"BGP MIB Support" module in the *IP Routing: BGP Configuration Guide*
Information about SNMP and SNMP operations	SNMP Configuration Guide

MIBs

MIB	MIBs Link
CISCO-BGP-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Cisco-BGP-MIBv2

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 19: Feature Information for Cisco-BGP-MIBv2

Feature Name	Releases	Feature Information
Cisco-BGP-MIBv2	15.2(1)E	The Cisco-BGP-MIBv2 feature supports CISCO-BGP-MIB, which allows MIBv2 traps to be generated. CISCO-BGP-MIB supports IPv4, IPv6, VPNv4, and VPNv6 address families. Its MIB objects and SNMP notifications can keep track of prefix counters and capabilities related to BGP sessions. The following command was modified: **snmp-server enable traps bgp**.