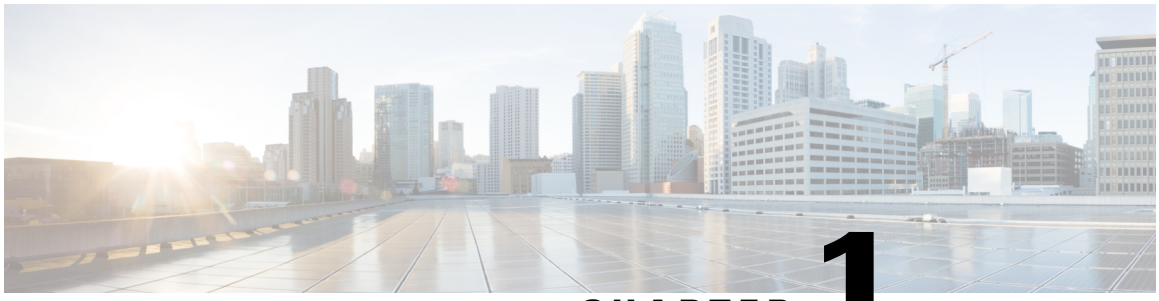
IP Routing: BGP Configuration Guide, Cisco IOS XE Release 3E

First Published: January 11, 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

Cisco BGP Overview

Border Gateway Protocol (BGP) is an interdomain routing protocol designed to provide loop-free routing between separate routing domains that contain independent routing policies (autonomous systems). The Cisco software implementation of BGP version 4 includes support for 4-byte autonomous system numbers and multiprotocol extensions to allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families including IP Version 4 (IPv4), IP Version 6 (IPv6), Virtual Private Networks Version 4 (VPNv4), Connectionless Network Services (CLNS), and Layer 2 VPN (L2VPN). This module contains conceptual material to help you understand how BGP is implemented in Cisco software.

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 Cisco BGP

This document assumes knowledge of CLNS, IPv4, IPv6, multicast, VPNv4, and Interior Gateway Protocols (IGPs). The amount of knowledge required for each technology is dependent on your deployment.

Restrictions for Cisco BGP

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

Information About Cisco BGP

BGP Version 4

Border Gateway Protocol (BGP) is an interdomain routing protocol designed to provide loop-free routing between separate routing domains that contain independent routing policies (autonomous systems). The Cisco software implementation of BGP version 4 includes multiprotocol extensions to allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families including IP Version 4 (IPv4), IP Version 6 (IPv6), Virtual Private Networks version 4 (VPNv4), and Connectionless Network Services (CLNS).

BGP is mainly used to connect a local network to an external network to gain access to the Internet or to connect to other organizations. When connecting to an external organization, external BGP (eBGP) peering sessions are created. For more details about connecting to external BGP peers, see the "Connecting to a Service Provider Using External BGP" chapter.

Although BGP is referred to as an exterior gateway protocol (EGP), many networks within an organization are becoming so complex that BGP can be used to simplify the internal network used within the organization. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions. For more details about internal BGP peers, see the "Configuring Internal BGP Features" chapter of the *Cisco IOS IP Routing Configuration Guide*.

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

BGP Version 4 Functional Overview

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

BGP is mainly used to connect a local network to an external network to gain access to the Internet or to connect to other organizations. When connecting to an external organization, external BGP (eBGP) peering

sessions are created. Although BGP is referred to as an exterior gateway protocol (EGP), many networks within an organization are becoming so complex that BGP can be used to simplify the internal network used within the organization. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

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

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

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

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

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

BGP Autonomous Systems

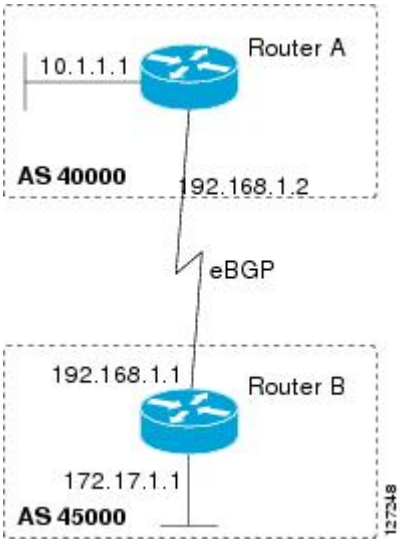
An autonomous system is a network controlled by a single technical administration entity. BGP autonomous systems are used to divide global external networks into individual routing domains where local routing policies are applied. This organization simplifies routing domain administration and simplifies consistent policy configuration. Consistent policy configuration is important to allow BGP to efficiently process routes to destination networks.

Each routing domain can support multiple routing protocols. However, each routing protocol is administered separately. Other routing protocols can dynamically exchange routing information with BGP through redistribution. Separate BGP autonomous systems dynamically exchange routing information through eBGP peering sessions. BGP peers within the same autonomous system exchange routing information through iBGP peering sessions.

The figure below illustrates two routers in separate autonomous systems that can be connected using BGP. Router A and Router B are ISP routers in separate routing domains that use public autonomous system numbers.

These routers carry traffic across the Internet. Router A and Router B are connected through eBGP peering sessions.

Figure 1: BGP Topology with Two Autonomous Systems



Each public autonomous system that directly connects to the Internet is assigned a unique number that identifies both the BGP routing process and the autonomous system.

BGP Autonomous System Number Formats

Prior to January 2009, BGP autonomous system numbers that were allocated to companies were two-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 Release 12.0(32)S12, 12.4(24)T, and later releases, 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 1: 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 Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, 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 2: 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 3: 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 Release 12.0(32)S12, 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, 12.4(24)T, 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/.

Classless Interdomain Routing

BGP version 4 supports classless interdomain routing (CIDR). CIDR eliminates classful network boundaries, providing more efficient usage of the IPv4 address space. CIDR provides a method to reduce the size of routing tables by configuring aggregate routes (or supernets). CIDR processes a prefix as an IP address and bit mask (bits are processed from left to right) to define each network. A prefix can represent a network, subnetwork, supernet, or single host route.

For example, using classful IP addressing, the IP address 192.168.2.1 is defined as a single host in the Class C network 192.168.2.0. Using CIDR, the IP address can be shown as 192.168.2.1/16, which defines a network (or supernet) of 192.168.0.0.

CIDR is enabled by default for all routing protocols in Cisco software. Enabling CIDR affects how packets are forwarded, but it does not change the operation of BGP.

Multiprotocol BGP

Cisco software supports multiprotocol BGP extensions as defined in RFC 2858, *Multiprotocol Extensions for BGP-4*. The extensions introduced in this RFC allow BGP to carry routing information for multiple network-layer protocols, including CLNS, IPv4, IPv6, and VPNv4. These extensions are backward-compatible to enable routers that do not support multiprotocol extensions to communicate with those routers that do support multiprotocol extensions. Multiprotocol BGP carries routing information for multiple network-layer protocols and IP multicast routes. BGP carries different sets of routes depending on the protocol. For example, BGP can carry one set of routes for IPv4 unicast routing, one set of routes for IPv4 multicast routing, and one set of routes for MPLS VPNv4 routes.

Note A multiprotocol BGP network is backward-compatible with a BGP network, but BGP peers that do not support multiprotocol extensions cannot forward routing information, such as address family identifier information, that the multiprotocol extensions carry.

Benefits of Using Multiprotocol BGP Versus BGP

In complex networks with multiple network layer protocols, multiprotocol BGP must be used. In less complex networks we recommend using multiprotocol BGP because it offers the following benefits:

- All of the BGP commands and routing policy capabilities of BGP can be applied to multiprotocol BGP.
- A network can carry routing information for multiple network layer protocol address families (for example, IP Version 4 or VPN Version 4) as specified in RFC 1700, *Assigned Numbers*.
- A network can support incongruent unicast and multicast topologies.
- A multiprotocol BGP network is backward compatible because the routers that support the multiprotocol extensions can interoperate with routers that do not support the extensions.

In summary, multiprotocol BGP support for multiple network layer protocol address families provides a flexible and scalable infrastructure that allows you to define independent policy and peering configurations on a per-address family basis.

Multiprotocol BGP Extensions for IP Multicast

The routes associated with multicast routing are used by the Protocol Independent Multicast (PIM) feature to build data distribution trees. Multiprotocol BGP is useful when you want a link that is dedicated to multicast traffic, perhaps to limit which resources are used for which traffic. For example, you want all multicast traffic exchanged at one network access point (NAP). Multiprotocol BGP allows you to have a unicast routing topology different from a multicast routing topology, which allows you more control over your network and resources.

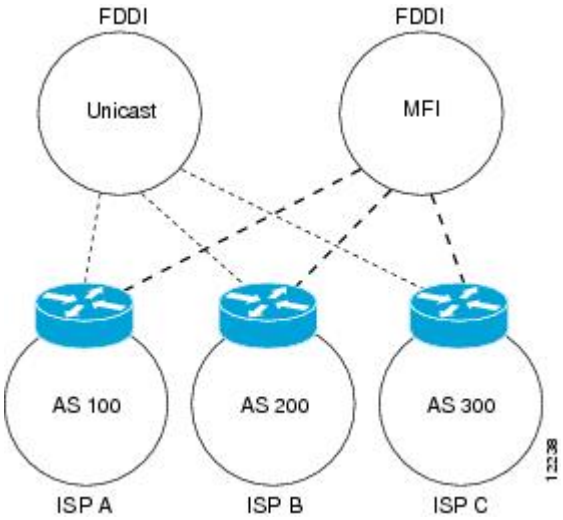
In BGP, the only way to perform interdomain multicast routing is to use the BGP infrastructure that is in place for unicast routing. If the routers are not multicast-capable, or if there are differing policies about where multicast traffic should flow, multicast routing cannot be supported without multiprotocol BGP.

A multicast routing protocol, such as PIM, uses both the multicast and unicast BGP database to source the route, perform Reverse Path Forwarding (RPF) lookups for multicast-capable sources, and build a multicast distribution tree (MDT). The multicast table is the primary source for the router, but if the route is not found in the multicast table, the unicast table is searched. Although multicast can be performed with unicast BGP, multicast BGP routes allow an alternative topology to be used for RPF.

It is possible to configure BGP peers that exchange both unicast and multicast Network Layer Reachability Information (NLRI) where multiprotocol BGP routes can be redistributed into BGP. Multiprotocol extensions, however, will be ignored by any peers that do not support multiprotocol BGP. When PIM builds a multicast distribution tree through a unicast BGP network (because the route through the unicast network is the most attractive), the RPF check may fail, preventing the MDT from being built. If the unicast network runs multiprotocol BGP, peering can be configured using the appropriate multicast address family. The multicast address family configuration enables multiprotocol BGP to carry the multicast information and the RPF lookup will succeed.

The figure below illustrates a simple example of unicast and multicast topologies that are incongruent; these topologies cannot exchange information without implementing multiprotocol BGP. Autonomous systems 100, 200, and 300 are each connected to two NAPs that are FDDI rings. One is used for unicast peering (and therefore the exchanging of unicast traffic). The Multicast Friendly Interconnect (MFI) ring is used for multicast peering (and therefore the exchanging of multicast traffic). Each router is unicast- and multicast-capable.

Figure 2: Incongruent Unicast and Multicast Routes



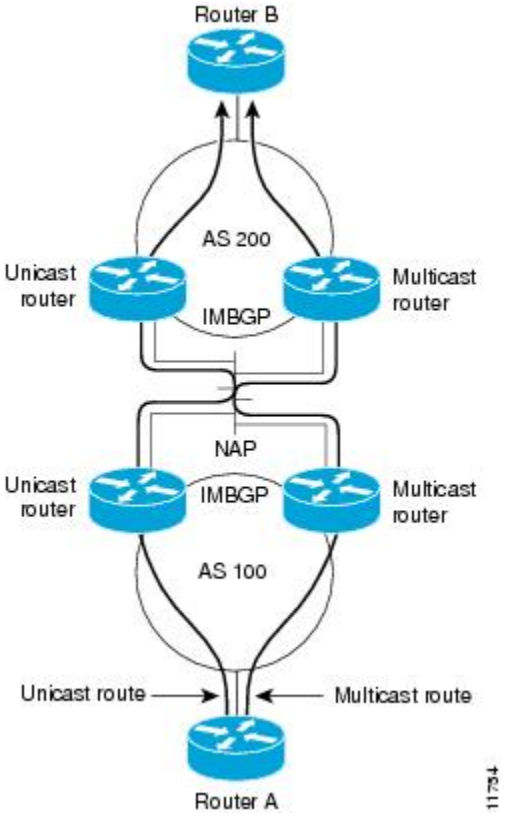
The figure below is a topology of unicast-only routers and multicast-only routers. The two routers on the left are unicast-only routers (that is, they do not support or are not configured to perform multicast routing). The two routers on the right are multicast-only routers. Routers A and B support both unicast and multicast routing. The unicast-only and multicast-only routers are connected to a single NAP.

In the figure below, only unicast traffic can travel from Router A to the unicast routers to Router B and back. Multicast traffic could not flow on that path, because multicast routing is not configured on the unicast routers and therefore the BGP routing table does not contain any multicast routes. On the multicast routers, multicast

routes are enabled and BGP builds a separate routing table to hold the multicast routes. Multicast traffic uses the path from Router A to the multicast routers to Router B and back.

The figure below illustrates a multiprotocol BGP environment with a separate unicast route and multicast route from Router A to Router B. Multiprotocol BGP allows these routes to be noncongruent. Both of the autonomous systems must be configured for internal multiprotocol BGP (labeled "IMBGP" in the figure).

Figure 3: Multicast BGP Environment



For more information about IP multicast, see the "Configuring IP Multicast" configuration library.

NLRI Configuration CLI

BGP was designed to carry only unicast IPv4 routing information. BGP configuration used the Network NLRI format CLI in Cisco software. The NLRI format offers only limited support for multicast routing information and does not support multiple network layer protocols. We do not recommend using NLRI format CLI for BGP configuration.

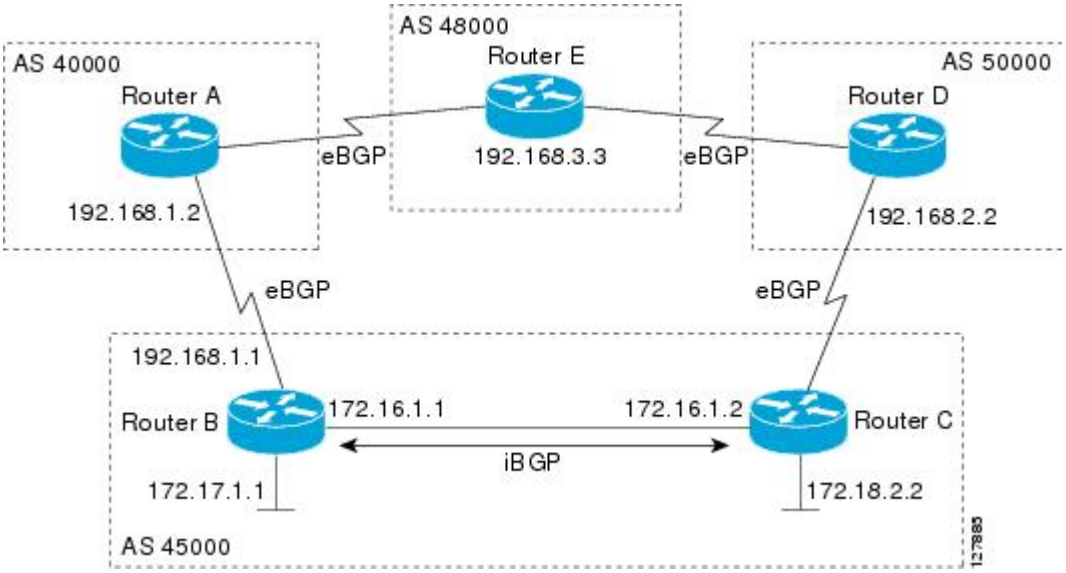
Using the BGP hybrid CLI feature, you can configure commands in the address family VPNv4 format and save these command configurations without modifying an existing NLRI formatted configuration. If you want to use other address family configurations such as IPv4 unicast or multicast, then you must upgrade the configuration using the **bgp upgrade-cli** command.

For more details about using BGP hybrid CLI commands, see the "Configuring a Basic BGP Network" module. See the "Multiprotocol BGP" and "Cisco BGP Address Family Model" sections for more information about address family configuration format and the limitations of the NLRI CLI format.

Cisco BGP Address Family Model

The Cisco BGP address family identifier (AFI) model was introduced with multiprotocol BGP and is designed to be modular and scalable, and to support multiple AFI and subsequent address family identifier (SAFI) configurations. Networks are increasing in complexity and many companies are now using BGP to connect to many autonomous systems, as shown in the network topology in the figure below. Each of the separate autonomous systems shown in the figure below may be running several routing protocols such as Multiprotocol Label Switching (MPLS) and IPv6 and require both unicast and multicast routes to be transported via BGP.

Figure 4: BGP Network Topology for Multiple Address Families



The Cisco BGP AFI model introduced new command-line interface (CLI) commands supported by a new internal structure. Multiprotocol BGP carries routing information for multiple network layer protocols and IP multicast routes. This routing information is carried in the AFI model as appended BGP attributes (multiprotocol extensions). Each address family maintains a separate BGP database, which allows you to configure BGP policy on per-address family basis. SAFI configurations are subsets of the parent AFI. SAFIs can be used to refine BGP policy configurations.

The AFI model was created because of scalability limitations of the NLRI format. A router that is configured in NLRI format has IPv4 unicast but limited multicast capabilities. Networks that are configured in the NLRI format have the following limitations:

- No support for AFI and SAFI configuration information. Many new BGP (and other protocols such as MPLS) features are supported only in AFI and SAFI configuration modes and cannot be configured in NLRI configuration modes.
- No support for IPv6. A router that is configured in the NLRI format cannot establish peering with an IPv6 neighbor.

- Limited support for multicast interdomain routing and incongruent multicast and unicast topologies. In the NLRI format, not all configuration options are available and there is no support for VPNv4. The NLRI format configurations can be more complex than configurations that support the AFI model. If the routers in the infrastructure do not have multicast capabilities, or if policies differ as to where multicast traffic is configured to flow, multicast routing cannot be supported.

The AFI model in multiprotocol BGP supports multiple AFIs and SAFIs, all NLRI-based commands and policy configurations, and is backward compatible with routers that support only the NLRI format. A router that is configured using the AFI model has the following features:

- AFI and SAFI information and configurations are supported. A router that is configured using the AFI model can carry routing information for multiple network layer protocol address families (for example, IPv4 and IPv6).
- AFI configuration is similar in all address families, making the CLI syntax easier to use than the NLRI format syntax.
- All BGP routing policy capabilities and commands are supported.
- Congruent unicast and multicast topologies that have different policies (BGP filtering configurations) are supported, as are incongruent multicast and unicast topologies.
- CLNS is supported.
- Interoperation between routers that support only the NLRI format (AFI-based networks are backward compatible) is supported. This includes both IPv4 unicast and multicast NLRI peers.
- Virtual Private Networks (VPNs) and VPN routing and forwarding (VRF) instances are supported. Unicast IPv4 for VRFs can be configured from a specific address family IPv4 VRF; this configuration update is integrated into the BGP VPNv4 database.

Within a specific address family configuration mode, the question mark (?) online help function can be used to display supported commands. The BGP commands supported in address family configuration mode configure the same functionality as the BGP commands supported in router configuration mode; however, the BGP commands in router configuration mode configure functionality only for the IPv4 unicast address prefix. To configure BGP commands and functionality for other address family prefixes (for example, the IPv4 multicast or IPv6 unicast address prefixes), you must enter address family configuration mode for those address prefixes.

The BGP address family model consists of four address families in Cisco IOS software; IPv4, IPv6, CLNS, and VPNv4. In Cisco IOS Release 12.2(33)SRB, and later releases, support for the L2VPN address family was introduced, and within the L2VPN address family the VPLS SAFI is supported. Within the IPv4 and IPv6 address families, SAFIs such as Multicast Distribution Tree (MDT), tunnel, and VRF exist. The table below shows the list of SAFIs supported by Cisco IOS software. To ensure compatibility between networks running all types of AFI and SAFI configuration, we recommend configuring BGP on Cisco IOS devices using the multiprotocol BGP address family model.

Table 4: SAFIs Supported by Cisco IOS Software

SAFI Field Value	Description	Reference
1	NLRI used for unicast forwarding.	RFC 2858
2	NLRI used for multicast forwarding.	RFC 2858

SAFI Field Value	Description	Reference
3	NLRI used for both unicast and multicast forwarding.	RFC 2858
4	NLRI with MPLS labels.	RFC 3107
64	Tunnel SAFI.	draft-nalawade-kapoor-tunnel- safi -01.txt
65	Virtual Private LAN Service (VPLS).	—
66	BGP MDT SAFI.	draft-nalawade-idr-mdt-safi-00.txt
128	MPLS-labeled VPN address.	RFC-ietf-l3vpn-rfc2547bis-03.txt

IPv4 Address Family

The IPv4 address family is used to identify routing sessions for protocols such as BGP that use standard IP version 4 address prefixes. Unicast or multicast address prefixes can be specified within the IPv4 address family. Routing information for address family IPv4 unicast is advertised by default when a BGP peer is configured unless the advertisement of unicast IPv4 information is explicitly turned off.

VRF instances can also be associated with IPv4 AFI configuration mode commands.

In Cisco IOS Release 12.0(28)S, the tunnel SAFI was introduced to support multipoint tunneling IPv4 routing sessions. The tunnel SAFI is used to advertise the tunnel endpoints and the SAFI specific attributes that contain the tunnel type and tunnel capabilities. Redistribution of tunnel endpoints into the BGP IPv4 tunnel SAFI table occurs automatically when the tunnel address family is configured. However, peers need to be activated under the tunnel address family before the sessions can exchange tunnel information.

In Cisco IOS Release 12.0(29)S, the multicast distribution tree (MDT) SAFI was introduced to support multicast VPN architectures. The MDT SAFI is a transitive multicast capable connector attribute that is defined as an IPv4 address family in BGP. The MDT address family session operates as a SAFI under the IPv4 multicast address family, and is configured on provider edge (PE) routers to establish VPN peering sessions with customer edge (CE) routers that support inter-AS multicast VPN peering sessions.

IPv6 Address Family

The IPv6 address family is used to identify routing sessions for protocols such as BGP that use standard IPv6 address prefixes. Unicast or multicast address prefixes can be specified within the IPv6 address family.

Note Routing information for address family IPv4 unicast is advertised by default when you configure a BGP peer unless you explicitly turn off the advertisement of unicast IPv4 information.

CLNS Address Family

The CLNS address family is used to identify routing sessions for protocols such as BGP that use standard network service access point (NSAP) address prefixes. Unicast address prefixes are the default when NSAP address prefixes are configured.

CLNS routes are used in networks where CLNS addresses are configured. This is typically a telecommunications Data Communications Network (DCN). Peering is established using IP addresses, but update messages contain CLNS routes.

For more details about configuring BGP support for CLNS, which provides the ability to scale CLNS networks, see the “Configuring Multiprotocol BGP (MP-BGP) support for CLNS” module.

VPNv4 Address Family

The VPNv4 multicast address family is used to identify routing sessions for protocols such as BGP that use standard VPN Version 4 address prefixes. Unicast address prefixes are the default when VPNv4 address prefixes are configured. VPNv4 routes are the same as IPv4 routes, but VPNv4 routes have a route descriptor (RD) prepended that allows replication of prefixes. It is possible to associate every different RD with a different VPN. Each VPN needs its own set of prefixes.

Companies use an IP VPN as the foundation for deploying or administering value-added services including applications and data hosting network commerce, and telephony services to business customers.

In private LANs, IP-based intranets have fundamentally changed the way companies conduct their business. Companies are moving their business applications to their intranets to extend over a WAN. Companies are also addressing the needs of their customers, suppliers, and partners by using extranets (an intranet that encompasses multiple businesses). With extranets, companies reduce business process costs by facilitating supply-chain automation, electronic data interchange (EDI), and other forms of network commerce. To take advantage of this business opportunity, service providers must have an IP VPN infrastructure that delivers private network services to businesses over a public infrastructure.

VPNs, when used with MPLS, allow several sites to transparently interconnect through a service provider's network. One service provider network can support several different IP VPNs. Each of these appears to its users as a private network, separate from all other networks. Within a VPN, each site can send IP packets to any other site in the same VPN. Each VPN is associated with one or more VPN VRFs. VPNv4 routes are a superset of routes from all VRFs, and route injection is done per VRF under the specific VRF address family. The router maintains a separate routing and Cisco Express Forwarding (CEF) table for each VRF. This prevents information from being sent outside the VPN and allows the same subnet to be used in several VPNs without causing duplicate IP address problems. The router using BGP distributes the VPN routing information using the BGP extended communities.

The VPN address space is isolated from the global address space by design. BGP distributes reachability information for VPN-IPv4 prefixes for each VPN using the VPNv4 multiprotocol extensions to ensure that the routes for a given VPN are learned only by other members of that VPN, enabling members of the VPN to communicate with each other.

RFC 3107 specifies how to add label information to multiprotocol BGP address families using a SAFI. The Cisco IOS implementation of MPLS uses RFC 3107 to provide support for sending IPv4 routes with a label. VPNv4 routes implicitly have a label associated with each route.

L2VPN Address Family

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 under 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.

Under L2VPN address family the following BGP command-line interface (CLI) commands are supported:

- **bgp scan-time**
- **bgp nexthop**
- **neighbor activate**
- **neighbor advertisement-interval**
- **neighbor allowas-in**
- **neighbor capability**
- **neighbor inherit**
- **neighbor peer-group**
- **neighbor maximum-prefix**
- **neighbor next-hop-self**
- **neighbor next-hop-unchanged**
- **neighbor remove-private-as**
- **neighbor route-map**
- **neighbor route-reflector-client**
- **neighbor send-community**
- **neighbor soft-reconfiguration**
- **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 under L2VPN address family configuration. All other route map commands are supported.

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

For details on configuring BGP under the L2VPN address family, see the "BGP Support for the L2VPN Address Family" module.

BGP CLI Removal Considerations

BGP CLI configuration can become quite complex even in smaller BGP networks. If you need to remove any CLI configuration, you must consider all the implications of removing the CLI. Analyze the current running configuration to determine the current BGP neighbor relationships, any address family considerations, and even other routing protocols that are configured. Many BGP CLI commands affect other parts of the CLI configuration. For example, in the following configuration, a route map is used to match a BGP autonomous system number and then set the matched routes with another autonomous system number for EIGRP:

```
route-map bgp-to-eigrp permit 10
 match tag 50000
 set tag 65000
```

BGP neighbors in three different autonomous systems are configured and activated:

```
router bgp 45000
 bgp log-neighbor-changes
 address-family ipv4
  neighbor 172.16.1.2 remote-as 45000
  neighbor 192.168.1.2 remote-as 40000
  neighbor 192.168.3.2 remote-as 50000
  neighbor 172.16.1.2 activate
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

An EIGRP routing process is then configured and BGP routes are redistributed into EIGRP with a route map filtering the routes:

```
router eigrp 100
 redistribute bgp 45000 metric 10000 100 255 1 1500 route-map bgp-to-eigrp
 no auto-summary
 exit
```

If you later decide to remove the route map, you will use the **no** form of the **route-map** command. Almost every configuration command has a **no** form, and the **no** form generally disables a function. However, in this configuration example, if you disable only the route map, the route redistribution will continue, but without the filtering or matching from the route map. Redistribution without the route map may cause unexpected behavior in your network. When you remove an access list or route map, you must also review the commands that referenced that access list or route map to consider whether the command will give you the behavior you intended.

The following configuration will remove both the route map and the redistribution:

```
configure terminal
```

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

For details on configuring the removal of BGP CLI configuration, see the "Configuring a Basic BGP Network" module.

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

Standard	Title
MDT SAFI	MDT SAFI

MIBs

MIB	MIBs Link
CISCO-BGP4-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1700	*Assigned Numbers*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 3107	*Carrying Label Information in BGP-4*
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*
RFC 4893	*BGP Support for Four-Octet AS Number Space*

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

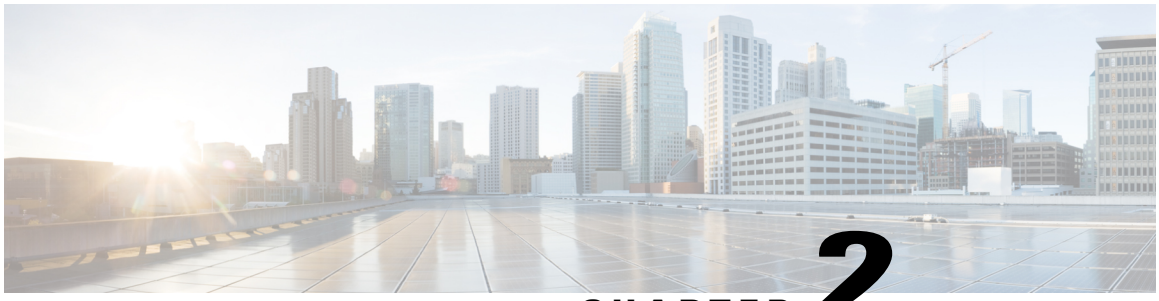
Feature Information for Cisco BGP Overview

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 5: Feature Information for Cisco BGP Overview

Feature Name	Releases	Feature Information
Multiprotocol BGP	Cisco IOS XE 3.1.0SG Cisco IOS XE Release 3.3SE Cisco IOS XE Release 3.6E	Cisco IOS software supports multiprotocol BGP extensions as defined in RFC 2858, *Multiprotocol Extensions for BGP-4* . The extensions introduced in this RFC allow BGP to carry routing information for multiple network layer protocols including CLNS, IPv4, IPv6, and VPNv4. These extensions are backward compatible to enable routers that do not support multiprotocol extensions to communicate with those routers that do support multiprotocol extensions. Multiprotocol BGP carries routing information for multiple network layer protocols and IP multicast routes. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.



CHAPTER 2

BGP 4

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

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 4

BGP Version 4 Functional Overview

BGP is an interdomain routing protocol designed to provide loop-free routing links between organizations. BGP is designed to run over a reliable transport protocol; it uses TCP (port 179) as the transport protocol because TCP is a connection-oriented protocol. The destination TCP port is assigned 179, and the local port is assigned a random port number. Cisco software supports BGP version 4 and it is this version that has been used by Internet service providers (ISPs) to help build the Internet. RFC 1771 introduced and discussed a

number of new BGP features to allow the protocol to scale for Internet use. RFC 2858 introduced multiprotocol extensions to allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families, including IPv4, IPv6, and CLNS.

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

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

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

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

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

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

BGP Router ID

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

BGP-Speaker and Peer Relationships

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

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

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

BGP Peer Session Establishment

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

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

peer. The hold timer restarts when an update or keepalive message is received. If the BGP process receives an error notification, it will transition to the Idle state.

BGP Session Reset

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

- Hard reset—A hard reset tears down the specified peering sessions including the TCP connection and deletes routes coming from the specified peer.
- Soft reset—A soft reset uses stored prefix information to reconfigure and activate BGP routing tables without tearing down existing peering sessions. Soft reconfiguration uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply new BGP policy without disrupting the network. Soft reconfiguration can be configured for inbound or outbound sessions.
- Dynamic inbound soft reset—The route refresh capability, as defined in RFC 2918, allows the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers. The route refresh capability does not store update information locally for nondisruptive policy changes. It instead relies on dynamic exchange with supporting peers. Route refresh must first be advertised through BGP capability negotiation between peers. All BGP devices must support the route refresh capability. To determine if a BGP device supports this capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the device supports the route refresh capability:

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

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

BGP Route Aggregation

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

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

The **bgp suppress-inactive** command configures BGP to not advertise inactive routes to any BGP peer. A BGP routing process can advertise routes that are not installed in the routing information database (RIB) to BGP peers by default. A route that is not installed into the RIB is an inactive route. Inactive route advertisement

can occur, for example, when routes are advertised through common route aggregation. Inactive route advertisements can be suppressed to provide more consistent data forwarding.

BGP Route Aggregation Generating AS_SET Information

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

Routing Policy Change Management

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

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

Table 6: Advantages and Disadvantages of Hard and Soft Resets

Type of Reset	Advantages	Disadvantages
Hard reset	No memory overhead.	The prefixes in the BGP, IP, and Forwarding Information Base (FIB) tables provided by the neighbor are lost. A hard reset is not recommended.
Outbound soft reset	No configuration, and no storing of routing table updates.	Does not reset inbound routing table updates.
Dynamic inbound soft reset	Does not clear the BGP session and cache. Does not require storing of routing table updates, and has no memory overhead.	Both BGP devices must support the route refresh capability. **Note** Does not reset outbound routing table updates.

Type of Reset	Advantages	Disadvantages
Configured inbound soft reset (uses the **neighbor soft-reconfiguration** router configuration command)	Can be used when both BGP devices do not support the automatic route refresh capability. The **bgp soft-reconfig-backup** command was introduced to configure inbound soft reconfiguration for peers that do not support the route refresh capability.	Requires preconfiguration. Stores all received (inbound) routing policy updates without modification; is memory-intensive. Recommended only when absolutely necessary, such as when both BGP devices do not support the automatic route refresh capability. **Note** Does not reset outbound routing table updates.

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

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

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

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

BGP Peer Groups

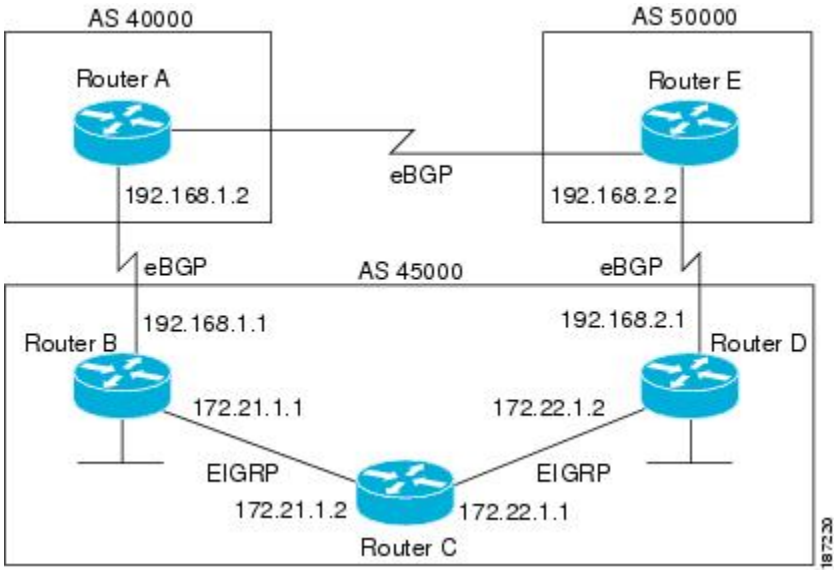
Often, in a BGP network, many neighbors are configured with the same update policies (that is, the same outbound route maps, distribute lists, filter lists, update source, and so on). Neighbors with the same update policies can be grouped into BGP peer groups to simplify configuration and, more importantly,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

BGP Backdoor Routes

In a BGP network topology with two border devices using eBGP to communicate to a number of different autonomous systems, using eBGP to communicate between the two border devices may not be the most efficient routing method. In the figure below, Router B as a BGP speaker will receive a route to Router D through eBGP, but this route will traverse at least two autonomous systems. Router B and Router D are also connected through an Enhanced Interior Gateway Routing Protocol (EIGRP) network (any IGP can be used here), and this route has a shorter path. EIGRP routes, however, have a default administrative distance of 90, and eBGP routes have a default administrative distance of 20, so BGP will prefer the eBGP route. Changing

the default administrative distances is not recommended because changing the administrative distance may lead to routing loops. To cause BGP to prefer the EIGRP route, you can use the **network backdoor** command. BGP treats the network specified by the **network backdoor** command as a locally assigned network, except that it does not advertise the specified network in BGP updates. In the figure below, this means that Router B will communicate to Router D using the shorter EIGRP route instead of the longer eBGP route.

Figure 5: BGP Backdoor Route Topology



How to Configure BGP 4

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

Configuring a BGP Routing Process

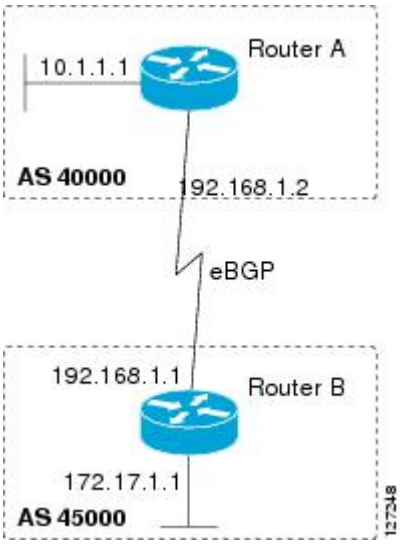
Perform this task to configure a BGP routing process. You must perform the required steps at least once to enable BGP. The optional steps here allow you to configure additional features in your BGP network. Several of the features, such as logging neighbor resets and immediate reset of a peer when its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

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

The configuration in this task is done at Router A in the figure below and would need to be repeated with appropriate changes to the IP addresses (for example, at Router B) to fully achieve a BGP process between

the two devices. No address family is configured here for the BGP routing process, so routing information for the IPv4 unicast address family is advertised by default.

Figure 6: BGP Topology with Two Autonomous Systems



SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
5. **bgp router-id** *ip-address*
6. **timers bgp** *keepalive holdtime*
7. **bgp fast-external-fallover**
8. **bgp log-neighbor-changes**
9. **end**
10. **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.

	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 40000`	Configures a BGP routing process, and enters router configuration mode for the specified routing process. • Use the *autonomous-system-number* argument to specify an integer, from 0 and 65534, that identifies the device to other BGP speakers.
Step 4	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router)# network 10.1.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols, the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 5	**bgp router-id** *ip-address* **Example:** `Device(config-router)# bgp router-id 10.1.1.99`	(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP. • Use the *ip-address* argument to specify a unique router ID within the network. **Note** Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.
Step 6	**timers bgp** *keepalive holdtime* **Example:** `Device(config-router)# timers bgp 70 120`	(Optional) Sets BGP network timers. • Use the *keepalive* argument to specify the frequency, in seconds, with which the software sends keepalive messages to its BGP peer. By default, the keepalive timer is set to 60 seconds. • Use the *holdtime* argument to specify the interval, in seconds, after which the software, having not received a keepalive message, declares a BGP peer dead. By default, the holdtime timer is set to 180 seconds.
Step 7	**bgp fast-external-fallover** **Example:** `Device(config-router)# bgp fast-external-fallover`	(Optional) Enables the automatic resetting of BGP sessions. • By default, the BGP sessions of any directly adjacent external peers are reset if the link used to reach them goes down.

	Command or Action	Purpose
Step 8	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	(Optional) Enables logging of BGP neighbor status changes (up or down) and neighbor resets. • Use this command for troubleshooting network connectivity problems and measuring network stability. Unexpected neighbor resets might indicate high error rates or high packet loss in the network and should be investigated.
Step 9	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 10	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

Troubleshooting Tips

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

Configuring a BGP Peer

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

Before You Begin

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

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** *ip-address* **activate**
7. **end**
8. **show ip bgp** [*network*] [*network-mask*]
9. **show ip bgp neighbors** [*neighbor-address*]

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 40000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

	Command or Action	Purpose
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.
Step 6	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local device.
Step 7	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 9	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device(config-router-af)# show ip bgp neighbors 192.168.2.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

```
BGP table version is 13, local router ID is 10.1.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
```

```
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      0.0.0.0                  0         32768 i
*> 172.17.1.0/24    192.168.1.1              0             0 45000 i
```

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

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

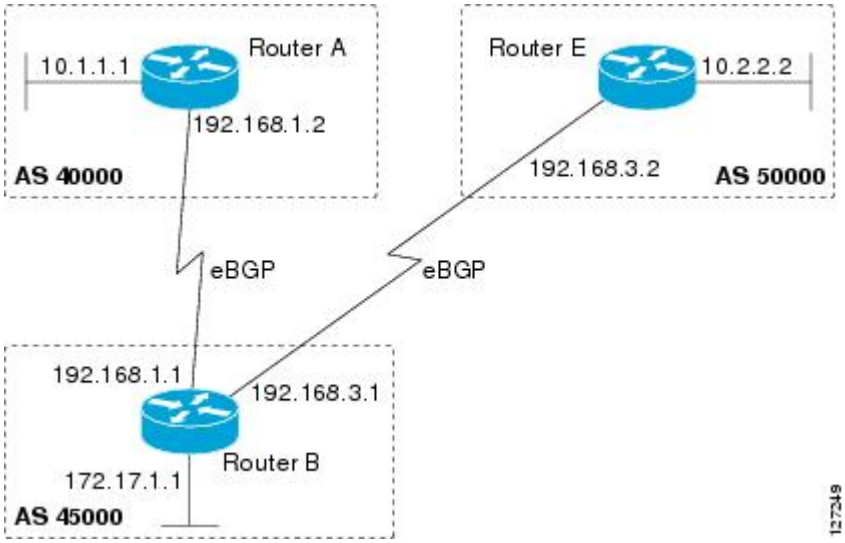
Troubleshooting Tips

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

Configuring a BGP Peer for the IPv4 VRF Address Family

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

Figure 7: BGP Topology for IPv4 VRF Address Family



Before You Begin

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **vrf forwarding** *vrf-name*
5. **ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]]
6. **exit**
7. **ip vrf** *vrf-name*
8. **rd** *route-distinguisher*
9. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
10. **exit**
11. **router bgp** *autonomous-system-number*
12. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**]
15. **neighbor** *ip-address* **activate**
16. **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	**interface** *type number*
	
	Example:
	`Device(config)# interface GigabitEthernet 0/0/0`
	
	`Device(config)# interface GigabitEthernet 0/0`
		Enters interface configuration mode.

	Command or Action	Purpose
Step 4	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding vpn1`	Associates a VPN VRF instance with an interface or subinterface.
Step 5	**ip address** *ip-address mask* [**secondary** [**vrf** *vrf-name*]] **Example:** `Device(config-if)# ip address 192.168.3.1 255.255.255.0`	Sets an IP address for an interface.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 7	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf vpn1`	Configures a VRF routing table and enters VRF configuration mode. • Use the *vrf-name* argument to specify a name to be assigned to the VRF.
Step 8	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 45000:5`	Creates routing and forwarding tables and specifies the default route distinguisher for a VPN. • Use the *route-distinguisher* argument to add an 8-byte value to an IPv4 prefix to create a unique VPN IPv4 prefix.
Step 9	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target both 45000:100`	Creates a route target extended community for a VRF. • Use the **import** keyword to import routing information from the target VPN extended community. • Use the **export** keyword to export routing information to the target VPN extended community. • Use the **both** keyword to import both import and export routing information to the target VPN extended community. • Use the *route-target-ext-community* argument to add the route target extended community attributes to the VRF's list of import, export, or both (import and export) route target extended communities.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 11	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 12	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 vrf vpn1`	Specifies the IPv4 address family and enters address family configuration mode. • Use the **unicast** keyword to specify the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • Use the **multicast** keyword to specify IPv4 multicast address prefixes. • Use the **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 14	**neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**] **Example:** `Device(config-router-af)# neighbor 192.168.3.2 maximum-prefix 10000 warning-only`	Controls how many prefixes can be received from a neighbor. • Use the *maximum* argument to specify the maximum number of prefixes allowed from the specified neighbor. The number of prefixes that can be configured is limited only by the available system resources on a device. • Use the *threshold* argument to specify an integer representing a percentage of the maximum prefix limit at which the device starts to generate a warning message. • Use the **warning-only** keyword to allow the device to generate a log message when the maximum prefix limit is exceeded, instead of terminating the peering session.

	Command or Action	Purpose
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local device.
Step 16	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

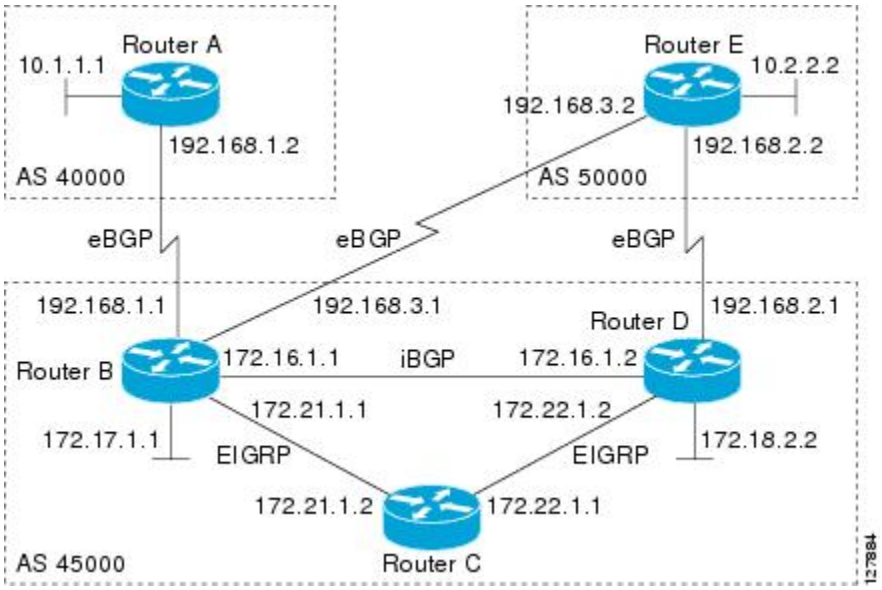
Use the **ping vrf** command to verify basic network connectivity between the BGP devices, and use the **show ip vrf** command to verify that the VRF instance has been created.

Customizing a BGP Peer

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

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

Figure 8: BGP Peer Topology



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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** {*ip-address* | *peer-group-name*} **description** *text*
7. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
8. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
9. **neighbor** {*ip-address* | *peer-group-name*} **activate**
10. **neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds*
11. **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]
12. **exit-address-family**
13. **neighbor** {*ip-address* | *peer-group-name*} **shutdown**
14. **end**
15. **show ip bgp ipv4 multicast** [*command*]
16. **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:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process.

	Command or Action	Purpose
		Note Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **description** *text* **Example:** `Device(config-router)# neighbor 192.168.3.2 description finance`	(Optional) Associates a text description with the specified neighbor.
Step 7	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 multicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 8	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.2 activate`	Enables the exchange of information with a BGP neighbor.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds* **Example:** `Device(config-router-af)# neighbor 192.168.3.2 advertisement-interval 25`	(Optional) Sets the minimum interval between the sending of BGP routing updates.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*] **Example:** `Device(config-router-af)# neighbor 192.168.3.2 default-originate`	(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.
Step 12	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters router configuration mode.
Step 13	**neighbor** {*ip-address* | *peer-group-name*} **shutdown** **Example:** `Device(config-router)# neighbor 192.168.3.2 shutdown`	(Optional) Disables a BGP peer or peer group. **Note** If you perform this step you will not be able to run either of the subsequent **show** command steps because you have disabled the neighbor.
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 15	**show ip bgp ipv4 multicast** [*command*] **Example:** `Device# show ip bgp ipv4 multicast`	(Optional) Displays IPv4 multicast database-related information. • Use the *command* argument to specify any multiprotocol BGP command that is supported. To see the supported commands, use the ? prompt on the CLI.
Step 16	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received prefix-filter**]	(Optional) Displays information about the TCP and BGP connections to neighbors.

	Command or Action	Purpose
	Example: `Device# show ip bgp neighbors 192.168.3.2`	

Examples

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

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

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

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

```
Local host: 192.168.3.1, Local port: 13172
Foreign host: 192.168.3.2, Foreign port: 179
!
```

Removing BGP Configuration Commands Using a Redistribution

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

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

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

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no route-map** *map-name*
4. **router eigrp** *autonomous-system-number*
5. **no redistribute** *protocol* [*as-number*]
6. **end**
7. **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:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**no route-map** *map-name* **Example:** `Device(config)# no route-map bgp-to-eigrp`	Removes a route map from the running configuration. • In this example, a route map named bgp-to-eigrp is removed from the configuration.
Step 4	**router eigrp** *autonomous-system-number* **Example:** `Device(config)# router eigrp 100`	Enters router configuration mode for the specified routing process.
Step 5	**no redistribute** *protocol* [*as-number*] **Example:** `Device(config-router)# no redistribute bgp 45000`	Disables the redistribution of routes from one routing domain into another routing domain. • In this example, the configuration of the redistribution of BGP routes into the EIGRP routing process is removed from the running configuration. **Note** If a route map was included in the original **redistribute** command configuration, remember to remove the **route-map** command configuration as in Step 3 in this example task. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	(Optional) Displays the current running configuration on the router. • Use this command to verify that the **redistribute** and **route-map** commands are removed from the router configuration.

Monitoring and Maintaining Basic BGP

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

Configuring Inbound Soft Reconfiguration When Route Refresh Capability Is Missing

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp log-neighbor-changes**
5. **bgp soft-reconfig-backup**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]
8. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
9. Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.
10. **exit**
11. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address*
13. **end**
14. **show ip bgp neighbors** [*neighbor-address*]
15. **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 for the specified routing process.

	Command or Action	Purpose
Step 4	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 5	**bgp soft-reconfig-backup** **Example:** `Device(config-router)# bgp soft-reconfig-backup`	Configures a BGP speaker to perform inbound soft reconfiguration for peers that do not support the route refresh capability. • This command is used to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**] **Example:** `Device(config-router)# neighbor 192.168.1.2 soft-reconfiguration inbound`	Configures the Cisco software to start storing updates. • All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is done later, the stored information will be used to generate a new set of inbound updates.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router)# neighbor 192.168.1.2 route-map LOCAL in`	Applies a route map to incoming or outgoing routes. • In this example, the route map named LOCAL will be applied to incoming routes.
Step 9	Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.	—
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LOCAL permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named LOCAL is created.
Step 12	**set ip next-hop** *ip-address* **Example:** `Device(config-route-map)# set ip next-hop 192.168.1.144`	Specifies where output packets that pass a match clause of a route map for policy routing. • In this example, the ip address is set to 192.168.1.144.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 15	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

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

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

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

```
BGP routing table entry for 172.17.1.0/24, version 11
Paths: (3 available, best #3, table Default-IP-Routing-Table, RIB-failure(4))
Flag: 0x820
 Advertised to update-groups:
    1
 50000
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external
 50000, (received-only)
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 100, valid, external
 40000
   192.168.1.2 from 192.168.1.2 (172.16.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external, best
```

Resetting and Displaying Basic BGP Information

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

SUMMARY STEPS

1. **enable**
2. **clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} [**soft** [**in** | **out**]]
3. **show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes** *mask-length*]
4. **show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*]
5. **show ip bgp paths**
6. **show ip bgp summary**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip bgp** {*****	*autonomous-system-number*	*neighbor-address*} [**soft** [**in**	**out**]] **Example:** `Device# clear ip bgp *`	Clears and resets BGP neighbor sessions: • In the example provided, all BGP neighbor sessions are cleared and reset.
Step 3	**show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name*	**route-map** *route-map-name*] [**shorter prefixes** *mask-length*]	Displays all the entries in the BGP routing table: • In the example provided, the BGP routing table information for the 10.1.1.0 network is displayed.

	Command or Action	Purpose
	Example: `Device#` **show ip bgp 10.1.1.0 255.255.255.0**	
Step 4	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*] **Example:** `Device#` **show ip bgp neighbors 192.168.3.2 advertised-routes**	Displays information about the TCP and BGP connections to neighbors. • In the example provided, the routes advertised from the device to BGP neighbor 192.168.3.2 on another device are displayed.
Step 5	**show ip bgp paths** **Example:** `Device#` **show ip bgp paths**	Displays information about all the BGP paths in the database.
Step 6	**show ip bgp summary** **Example:** `Device#` **show ip bgp summary**	Displays information about the status of all BGP connections.

Aggregating Route Prefixes Using BGP

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

Redistributing a Static Aggregate Route into BGP

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]
4. **router bgp** *autonomous-system-number*
5. **redistribute static**
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	**ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*] **Example:** `Device(config)# ip route 172.0.0.0 255.0.0.0 null 0`	Creates a static route.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 5	**redistribute static** **Example:** `Device(config-router)# redistribute static`	Redistributes routes into the BGP routing table.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

Configuring Conditional Aggregate Routes Using BGP

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **aggregate-address** *address mask* [**as-set**]
5. **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** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**aggregate-address** *address mask* [**as-set**] **Example:** `Device(config-router)# aggregate-address 172.0.0.0 255.0.0.0 as-set`	Creates an aggregate entry in a BGP routing table. • A specified route must exist in the BGP table. • Use the **aggregate-address** command with no keywords to create an aggregate entry if any more-specific BGP routes are available that fall in the specified range. • Use the **as-set** keyword to specify that the path advertised for this route is an AS_SET. Do not use the **as-set** keyword when aggregating many paths because this route is withdrawn and updated

	Command or Action	Purpose
		every time the reachability information for the aggregated route changes. **Note** Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 5	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Suppressing and Unsuppressing the Advertisement of Aggregated Routes Using BGP

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. Do one of the following:
   - **aggregate-address** *address mask* [**summary-only**]
   - **aggregate-address** *address mask* [**suppress-map** *map-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **unsuppress-map** *map-name*
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** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

Step 5	Do one of the following:
	• **aggregate-address** *address mask* [**summary-only**]
	• **aggregate-address** *address mask* [**suppress-map** *map-name*]
	
	Example:
	`Device(config-router)# aggregate-address 172.0.0.0 255.0.0.0 summary-only`
	
	Example:
	`Device(config-router)# aggregate-address 172.0.0.0 255.0.0.0 suppress-map map1`
		Creates an aggregate route.
		• Use the optional **summary-only** keyword to create the aggregate route (for example, 10.*.*.*) and also suppresses advertisements of more-specific routes to all neighbors.
		• Use the optional **suppress-map** keyword to create the aggregate route but suppress advertisement of specified routes. Routes that are suppressed are not advertised to any neighbors. You can use the **match** clauses of route maps to selectively suppress some more-specific routes of the aggregate and leave others unsuppressed. IP access lists and autonomous system path access lists **match** clauses are supported.
		
		Note Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Step 6	**neighbor** {*ip-address* | *peer-group-name*} **unsuppress-map** *map-name* **Example:** `Device(config-router)# neighbor 192.168.1.2 unsuppress map1`	(Optional) Selectively advertises routes previously suppressed by the **aggregate-address** command. • In this example, the routes previously suppressed in Step 5 are advertised to neighbor 192.168.1.2.
Step 7	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Conditionally Advertising BGP Routes

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

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

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

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* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*}
6. **exit**
7. **route-map** *map-tag* [**permit** | **deny**] [**sequence-number**]
8. **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}
9. **exit**
10. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
11. **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}
12. **exit**
13. **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]
14. **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]
15. **exit**

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 for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 5	**neighbor** *ip-address* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*} **Example:** `Device(config-router)# neighbor 192.168.1.2 advertise-map map1 exist-map map2`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device. • In this example, the prefix (172.17.0.0) matching the ACL in the advertise map (the route map named map1) will be advertised to the neighbor only when a prefix (192.168.50.0) matching the ACL in exist map (the route-map named map2) is in the local BGP table.
Step 6	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 7	**route-map** *map-tag* [**permit** | **deny**] [**sequence-number**] **Example:** `Device(config)# route-map map1 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named map1 is created.
Step 8	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* |	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list.

	Command or Action	Purpose
	access-list-name] | **prefix-list** *prefix-list-name* [*prefix-list-name*...]} **Example:** `Device(config-route-map)# match ip address 1`	• In this example, the route map is configured to match a prefix permitted by access list 1.
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 10	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map map2 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named map2 is created.
Step 11	**match ip address** {*access-list-number* [*access-list-number*... | *access-list-name*...] | *access-list-name* [*access-list-number*... | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name*...]} **Example:** `Device(config-route-map)# match ip address 2`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by access list 2.
Step 12	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 13	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 1 permit 172.17.0.0`	Configures a standard access list. • In this example, access list 1 permits advertising of the 172.17.0.0 prefix, depending on other conditions set by the **neighbor advertise-map** command.
Step 14	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 2 permit 192.168.50.0`	Configures a standard access list. • In this example, access list 2 permits the 192.168.50.0 to be the prefix of the exist-map.

	Command or Action	Purpose
Step 15	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Originating BGP Routes

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

Advertising a Default Route Using BGP

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network / length* | **permit** *network / length*} [**ge** *ge-value*] [**le** *le-value*]
4. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
5. **match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}
6. **exit**
7. **router bgp** *autonomous-system-number*
8. **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*]
9. **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	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network / length* | **permit** *network / length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list DEFAULT permit 10.1.1.0/24`	Configures an IP prefix list. • In this example, prefix list DEFAULT permits advertising of the 10.1.1.0/24. prefix depending on a match set by the **match ip address** command.
Step 4	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map ROUTE`	Configures a route map and enters route map configuration mode. • In this example, a route map named ROUTE is created.
Step 5	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# match ip address prefix-list DEFAULT`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by prefix list DEFAULT.
Step 6	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 7	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*] **Example:** `Device(config-router)# neighbor 192.168.3.2 default-originate`	(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.
Step 9	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Originating BGP Routes Using Backdoor Routes

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

Before You Begin

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. **network** *ip-address* **backdoor**
6. **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** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 172.22.1.2 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device. • In this example, the peer is an internal peer as the autonomous system number specified for the peer is the same number specified in Step 3.
Step 5	**network** *ip-address* **backdoor** **Example:** `Device(config-router)# network 172.21.1.0 backdoor`	Indicates a network that is reachable through a backdoor route.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

Configuring a BGP Peer Group

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

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

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

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

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **peer-group** *peer-group-name*
7. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
8. **neighbor** *peer-group-name* **activate**
9. **neighbor** *ip-address* **peer-group** *peer-group-name*
10. **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** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor fingroup peer-group`	Creates a BGP peer group.
Step 5	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the multiprotocol BGP neighbor table of the local device.
Step 6	**neighbor** *ip-address* **peer-group** *peer-group-name* **Example:** `Device(config-router)# neighbor 192.168.1.1 peer-group fingroup`	Assigns the IP address of a BGP neighbor to a peer group.
Step 7	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 multicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. This is the default. • The **multicast** keyword specifies that IPv4 multicast address prefixes will be exchanged. • The **vrf** keyword and *vrf-name* argument specify that IPv4 VRF instance information will be exchanged.
Step 8	**neighbor** *peer-group-name* **activate** **Example:** `Device(config-router-af)# neighbor fingroup activate`	Enables the neighbor to exchange prefixes for the IPv4 address family with the local device. **Note** By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only unicast address prefixes. To allow BGP to exchange other address prefix types, such as multicast that is configured in this example, neighbors must also be activated using the **neighbor activate** command.
Step 9	**neighbor** *ip-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 peer-group fingroup`	Assigns the IP address of a BGP neighbor to a peer group.

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

Configuration Examples for BGP 4

Example: Configuring a BGP Process and Customizing Peers

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

Router B

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

Examples: Removing BGP Configuration Commands Using a Redistribution Example

The following examples show first the CLI configuration to enable the redistribution of BGP routes into EIGRP using a route map and then the CLI configuration to remove the redistribution and route map. Some

BGP configuration commands can affect other CLI commands and this example demonstrates how the removal of one command affects another command.

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

CLI to Enable BGP Route Redistribution Into EIGRP

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

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

CLI to Remove BGP Route Redistribution Into EIGRP

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

Examples: BGP Soft Reset

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

Example: Dynamic Inbound Soft Reset

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

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

Example: Inbound Soft Reset Using Stored Information

The following example shows how to enable inbound soft reconfiguration for the neighbor 192.168.1.1. All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When

inbound soft reconfiguration is performed later, the stored information will be used to generate a new set of inbound updates.

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

The following example clears the session with the neighbor 192.168.1.1:

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

Example: Resetting and Displaying Basic BGP Information

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

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

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

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

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

```
Device# show ip bgp 10.1.1.0 255.255.255.0

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

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

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

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

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

```
Device# show ip bgp paths

Address    Hash Refcount Metric Path
0x2FB5DB0     0        5      0 i
```

```
0x2FB5C90     1        4      0 i
0x2FB5C00  1361        2      0 50000 i
0x2FB5D20  2625        2      0 40000 i
```

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

```
Device# show ip bgp summary

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

Examples: Aggregating Prefixes Using BGP

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

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

```
ip route 10.0.0.0 255.0.0.0 null 0
!
router bgp 100
 redistribute static
```

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

```
router bgp 100
 aggregate-address 10.0.0.0 255.0.0.0
```

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

```
router bgp 100
 aggregate-address 10.0.0.0 255.0.0.0 as-set
```

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

```
router bgp 100
 aggregate-address 10.0.0.0 255.0.0.0 summary-only
```

The following example configures BGP to not advertise inactive routes:

```
Device(config)# router bgp 50000
Device(config-router)# address-family ipv4 unicast
```

```
Device(config-router-af)# bgp suppress-inactive
Device(config-router-af)# end
```

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

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

Example: Configuring a BGP Peer Group

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

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

Additional References

Related Documents

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

Standards and RFCs

Standard/RFC	Title
RFC 1772	*Application of the Border Gateway Protocol in the Internet*
RFC 1773	*Experience with the BGP Protocol*
RFC 1774	*BGP-4 Protocol Analysis*
RFC 1930	*Guidelines for Creation, Selection, and Registration on an Autonomous System (AS)*
RFC 2519	*A Framework for Inter-Domain Route Aggregation*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 2918	*Route Refresh Capability for BGP-4*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*

MIBs

MIB	MIBs Link
CISCO-BGP4-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

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

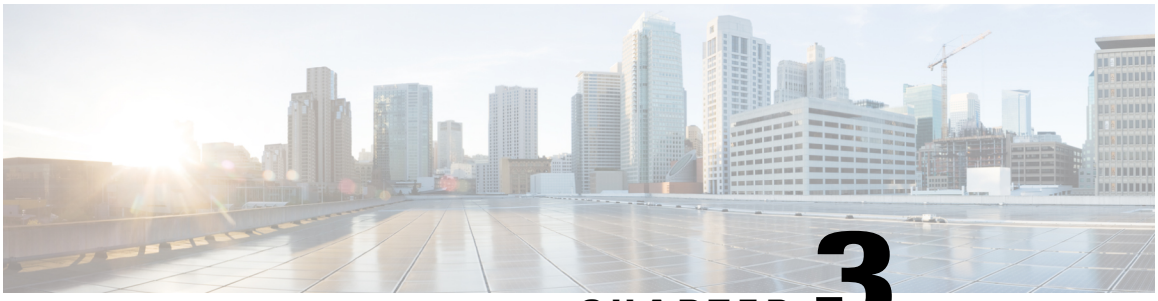
Feature Information for BGP 4

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 4

Feature Name	Releases	Feature Information
BGP 4	Cisco IOS XE Release 3.3SE Cisco IOS XE Release 3.6E	BGP is an interdomain routing protocol designed to provide loop-free routing between separate routing domains that contain independent routing policies (autonomous systems). The Cisco software implementation of BGP Version 4 includes multiprotocol extensions to allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families, including IP Version 4 (IPv4), IP Version 6 (IPv6), Virtual Private Networks version 4 (VPNv4), and Connectionless Network Services (CLNS). In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.



CHAPTER 3

BGP NSF Awareness

Nonstop Forwarding (NSF) awareness allows a device to assist NSF-capable neighbors to continue forwarding packets during a Stateful Switchover (SSO) operation. The BGP NSF Awareness feature allows an NSF-aware device that is running BGP to forward packets along routes that are already known for a device that is performing an SSO operation. This capability allows the BGP peers of the failing device to retain the routing information that is advertised by the failing device and continue to use this information until the failed device has returned to normal operating behavior and is able to exchange routing information. The peering session is maintained throughout the entire NSF operation.

Finding Feature Information

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

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

Restrictions for BGP NSF Awareness

On Cisco Catalyst 3650 Series Switches, virtual routing and forwarding (VRF) instances are not supported for the BGP NSF Awareness feature.

Information About BGP NSF Awareness

Cisco NSF Routing and Forwarding Operation

Cisco NSF is supported by the BGP, EIGRP, OSPF, and IS-IS protocols for routing and by Cisco Express Forwarding (CEF) for forwarding. Of the routing protocols, BGP, EIGRP, OSPF, and IS-IS have been enhanced with NSF capability and awareness, which means that devices running these protocols can detect a switchover and take the necessary actions to continue forwarding network traffic and to recover route information from the peer devices.

In this module, a networking device is said to be NSF-aware if it is running NSF-compatible software. A device is said to be NSF-capable if it has been configured to support NSF; therefore, it rebuilds routing information from NSF-aware or NSF-capable neighbors.

Each protocol depends on CEF to continue forwarding packets during switchover while the routing protocols rebuild the Routing Information Base (RIB) tables. Once the routing protocols have converged, CEF updates the Forwarding Information Base (FIB) table and removes stale route entries. CEF then updates the line cards with the new FIB information.

Cisco Express Forwarding for NSF

A key element of NSF is packet forwarding. In a Cisco networking device, packet forwarding is provided by CEF. CEF maintains the FIB and uses the FIB information that was current at the time of the switchover to continue forwarding packets during a switchover. This feature reduces traffic interruption during the switchover.

During normal NSF operation, CEF on the active RP synchronizes its current FIB and adjacency databases with the FIB and adjacency databases on the standby RP. Upon switchover of the active RP, the standby RP initially has FIB and adjacency databases that are mirror images of those that were current on the active RP. For platforms with intelligent line cards, the line cards will maintain the current forwarding information over a switchover; for platforms with forwarding engines, CEF will keep the forwarding engine on the standby RP current with changes that are sent to it by CEF on the active RP. In this way, the line cards or forwarding engines will be able to continue forwarding after a switchover as soon as the interfaces and a data path are available.

As the routing protocols start to repopulate the RIB on a prefix-by-prefix basis, the updates in turn cause prefix-by-prefix updates for CEF, which it uses to update the FIB and adjacency databases. Existing and new entries will receive the new version (epoch) number, indicating that they have been refreshed. The forwarding information is updated on the line cards or forwarding engine during convergence. The RP signals when the RIB has converged. The software removes all FIB and adjacency entries that have an epoch older than the current switchover epoch. The FIB now represents the newest routing protocol forwarding information.

The routing protocols run only on the active RP, and they receive routing updates from their neighbor routers. Routing protocols do not run on the standby RP. After a switchover, the routing protocols request that the NSF-aware neighbor devices send state information to help rebuild the routing tables.

Note For NSF operation, the routing protocols depend on CEF to continue forwarding packets while the routing protocols rebuild the routing information.

BGP Graceful Restart for NSF

When an NSF-capable router begins a BGP session with a BGP peer, it sends an OPEN message to the peer. Included in the message is a declaration that the NSF-capable or NSF-aware router has graceful restart capability. Graceful restart is the mechanism by which BGP routing peers avoid a routing flap after a switchover. If the BGP peer has received this capability, it is aware that the device sending the message is NSF-capable. Both the NSF-capable router and its BGP peer(s) (NSF-aware peers) need to exchange the graceful restart capability in their OPEN messages, at the time of session establishment. If both peers do not exchange the graceful restart capability, the session will not be graceful restart capable.

If the BGP session is lost during the RP switchover, the NSF-aware BGP peer marks all the routes associated with the NSF-capable router as stale; however, it continues to use these routes to make forwarding decisions for a set period of time. This functionality means that no packets are lost while the newly active RP is waiting for convergence of the routing information with the BGP peers.

After an RP switchover occurs, the NSF-capable router reestablishes the session with the BGP peer. In establishing the new session, it sends a new graceful restart message that identifies the NSF-capable router as having restarted.

At this point, the routing information is exchanged between the two BGP peers. Once this exchange is complete, the NSF-capable device uses the routing information to update the RIB and the FIB with the new forwarding information. The NSF-aware device uses the network information to remove stale routes from its BGP table. Following that, the BGP protocol is fully converged.

If a BGP peer does not support the graceful restart capability, it will ignore the graceful restart capability in an OPEN message but will establish a BGP session with the NSF-capable device. This functionality will allow interoperability with non-NSF-aware BGP peers (and without NSF functionality), but the BGP session with non-NSF-aware BGP peers will not be graceful restart capable.

BGP NSF Awareness

BGP support for NSF requires that neighbor routers are NSF-aware or NSF-capable. NSF awareness in BGP is also enabled by the graceful restart mechanism. A router that is NSF-aware functions like a router that is NSF-capable with one exception: an NSF-aware router is incapable of performing an SSO operation. However, a router that is NSF-aware is capable of maintaining a peering relationship with an NSF-capable neighbor during an NSF SSO operation, as well as holding routes for this neighbor during the SSO operation.

The BGP Nonstop Forwarding Awareness feature provides an NSF-aware router with the capability to detect a neighbor that is undergoing an SSO operation, maintain the peering session with this neighbor, retain known routes, and continue to forward packets for these routes. The deployment of BGP NSF awareness can minimize the effects of Route Processor (RP) failure conditions and improve the overall network stability by reducing the amount of resources that are normally required for reestablishing peering with a failed router.

NSF awareness for BGP is not enabled by default. The **bgp graceful-restart** command is used to globally enable NSF awareness on a router that is running BGP. NSF-aware operations are also transparent to the network operator and to BGP peers that do not support NSF capabilities.

Note NSF awareness is enabled automatically in supported software images for Interior Gateway Protocols, such as EIGRP, IS-IS, and OSPF. In BGP, global NSF awareness is not enabled automatically and must be started by issuing the **bgp graceful-restart** command in router configuration mode.

How to Configure BGP NSF Awareness

Configuring BGP Nonstop Forwarding Awareness Using BGP Graceful Restart

The tasks in this section show how configure BGP Nonstop Forwarding (NSF) awareness using the BGP graceful restart capability.

- The first task enables BGP NSF globally for all BGP neighbors and suggests a few troubleshooting options.
- The second task describes how to adjust the BGP graceful restart timers, although the default settings are optimal for most network deployments.
- The next three tasks demonstrate how to enable or disable BGP graceful restart for individual BGP neighbors, including peer session templates and peer groups.
- The final task verifies the local and peer router configurations of BGP NSF.

Enabling BGP Global NSF Awareness Using BGP Graceful Restart

Perform this task to enable BGP NSF awareness globally for all BGP neighbors. BGP NSF awareness is part of the graceful restart mechanism and BGP NSF awareness is enabled by issuing the **bgp graceful-restart** command in router configuration mode. BGP NSF awareness allows NSF-aware routers to support NSF-capable routers during an SSO operation. NSF-awareness is not enabled by default and should be configured on all neighbors that participate in BGP NSF.

Note The configuration of the restart and stale-path timers is not required to enable the BGP graceful restart capability. The default values are optimal for most network deployments, and these values should be adjusted only by an experienced network operator.

Note Configuring both Bidirectional Forwarding Detection (BFD) and BGP graceful restart for NSF on a device running BGP may result in suboptimal routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp graceful-restart** [**restart-time** *seconds*] [**stalepath-time** *seconds*]
5. **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** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**bgp graceful-restart** [**restart-time** *seconds*] [**stalepath-time** *seconds*] **Example:** `Device(config-router)# bgp graceful-restart`	Enables the BGP graceful restart capability and BGP NSF awareness. • 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. • Use this command on the restarting router and all of its peers (NSF-capable and NSF-aware).
Step 5	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

To troubleshoot the NSF feature, use the following commands in privileged EXEC mode, as needed:

- **debug ip bgp —**Displays open messages that advertise the graceful restart capability.
- **debug ip bgp event —**Displays graceful restart timer events, such as the restart timer and the stalepath timer.
- **debug ip bgp updates —**Displays sent and received EOR messages. The EOR message is used by the NSF-aware router to start the stalepath timer, if configured.
- **show ip bgp —**Displays entries in the BGP routing table. The output from this command displays routes that are marked as stale by displaying the letter “S” next to each stale route.

- **show ip bgp neighbor —**Displays information about the TCP and BGP connections to neighbor devices. When enabled, the graceful restart capability is displayed in the output of this command.

What to Do Next

After the peer session template is created, the configuration of the peer session template can be inherited or applied by another peer session template with the **inherit peer-session** or **neighbor inherit peer-session** command.

Configuring BGP NSF Awareness Timers

Perform this task to adjust the BGP graceful restart timers. There are two BGP graceful restart timers that can be configured. The optional **restart-time** keyword and *seconds* argument determine how long peer routers will wait to delete stale routes before a BGP open message is received. The default value is 120 seconds. The optional **stalepath-time** keyword and *seconds* argument determine how long a router will wait before deleting stale routes after an end of record (EOR) message is received from the restarting router. The default value is 360 seconds.

Note The configuration of the restart and stale-path timers is not required to enable the BGP graceful restart capability. The default values are optimal for most network deployments, and these values should be adjusted only by an experienced network operator.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp graceful-restart** [**restart-time** *seconds*]
5. **bgp graceful-restart** [**stalepath-time** *seconds*]
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** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**bgp graceful-restart** [**restart-time** *seconds*] **Example:** `Device(config-router)# bgp graceful-restart restart-time 130`	Enables the BGP graceful restart capability and BGP NSF awareness. • The **restart-time** argument determines how long peer routers will wait to delete stale routes before a BGP open message is received. • The default value is 120 seconds. The range is from 1 to 3600 seconds. **Note** Only the syntax applicable to this step is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 5	**bgp graceful-restart** [**stalepath-time** *seconds*] **Example:** `Device(config-router)# bgp graceful-restart stalepath-time 350`	Enables the BGP graceful restart capability and BGP NSF awareness. • The **stalepath-time** argument determines how long a router will wait before deleting stale routes after an end of record (EOR) message is received from the restarting router. • The default value is 360 seconds. The range is from 1 to 3600 seconds. **Note** Only the syntax applicable to this step is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

What to Do Next

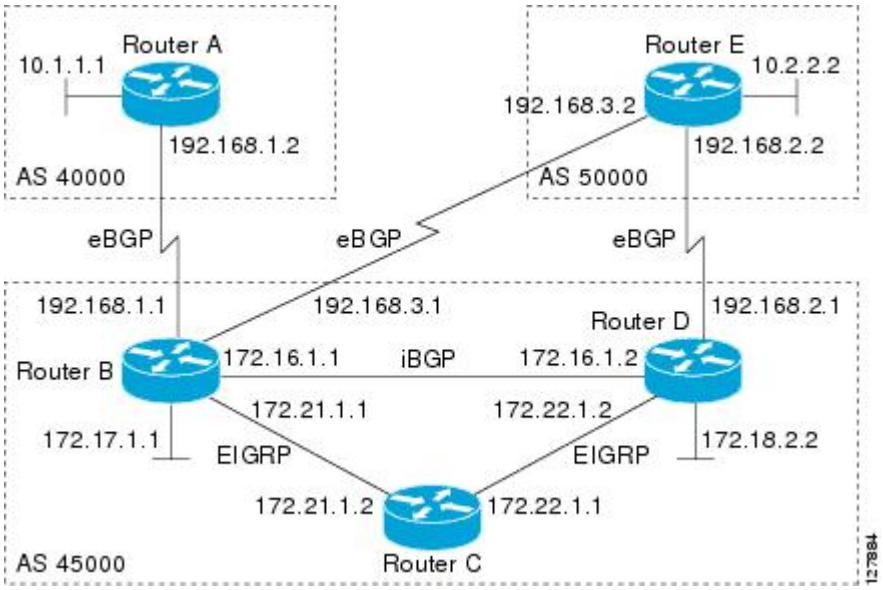
If the **bgp graceful-restart** command has been issued after the BGP session has been established, you must reset by issuing the **clear ip bgp *** command or by reloading the router before graceful restart capabilities will be exchanged. For more information about resetting BGP sessions and using the **clear ip bgp** command, see the "Configuring a Basic BGP Network" module.

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 9: 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.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.

	Command or Action	Purpose
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.
Step 9	**exit-peer-session** **Example:** `Device(config-router-stmp)# exit-peer-session`	Exits session-template configuration mode and returns to router configuration mode.

	Command or Action	Purpose
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.
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*]	(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.

	Command or Action	Purpose
	Example: `Device# show ip bgp template peer-session`	
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
```

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.

	Command or Action	Purpose
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** *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**]]]	(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.

	Command or Action	Purpose
	Example: `Device# show ip bgp neighbors 172.21.1.2`	• 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
```

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.
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** *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.
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
```

Verifying the Configuration of BGP Nonstop Forwarding Awareness

Use the following steps to verify the local configuration of BGP NSF awareness on a router and to verify the configuration of NSF awareness on peer routers in a BGP network.

SUMMARY STEPS

1. **enable**
2. **show running-config** [*options*]
3. **show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show running-config** [*options*]

Displays the running configuration on the local router. The output will display the configuration of the **bgp graceful-restart** command in the BGP section. Repeat this command on all BGP neighbor routers to verify that all BGP peers are configured for BGP NSF awareness. In this example, BGP graceful restart is enabled globally and the external neighbor at 192.168.1.2 is configured to be a BGP peer and will have the BGP graceful restart capability enabled.

Example:

```
Device# show running-config
.
.
.
router bgp 45000
```

```
 bgp router-id 172.17.1.99
 bgp log-neighbor-changes
 bgp graceful-restart restart-time 130
 bgp graceful-restart stalepath-time 350
 bgp graceful-restart
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.1.2 activate
.
.
.
```

Step 3 **show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]

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.

Configuration Examples for BGP NSF Awareness

Example: Enabling BGP Global NSF Awareness Using Graceful Restart

The following example enables BGP NSF awareness globally on all BGP neighbors. The restart time is set to 130 seconds, and the stale path time is set to 350 seconds. The configuration of these timers is optional, and the preconfigured default values are optimal for most network deployments.

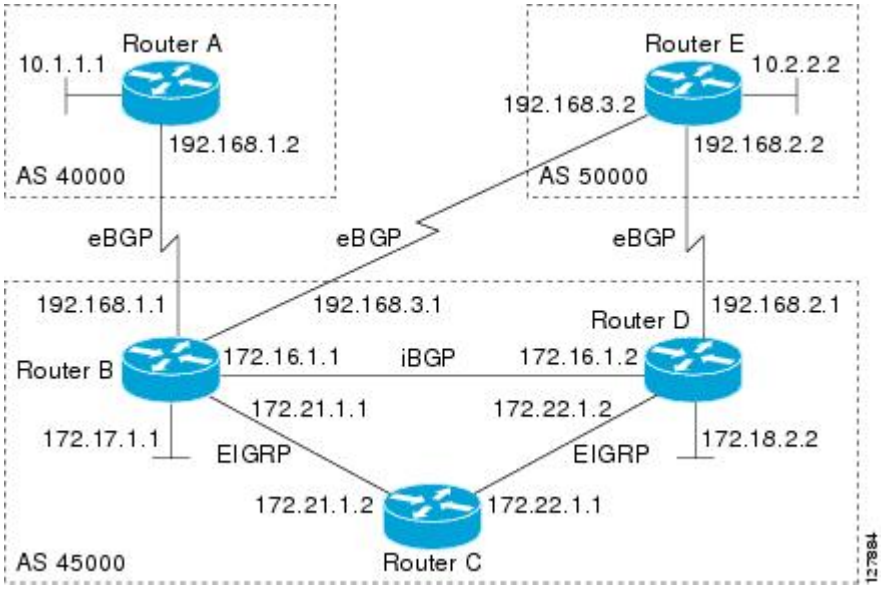
```
configure terminal
 router bgp 45000
 bgp graceful-restart
 bgp graceful-restart restart-time 130
 bgp graceful-restart stalepath-time 350
 end
```

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 10: 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
Enabling BGP MIB support	"BGP MIB Support" module in the *IP Routing: BGP Configuration Guide*
Configuring SNMP Support	*SNMP Configuration Guide* in the *Cisco IOS Network Management Configuration Guide Library*
SNMP Commands	*Cisco IOS SNMP Support Command Reference*

Standards

Standard	Title
None	—

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
RFC 1657	*BGP-4 MIB*
RFC 1771	*A Border Gateway Protocol 4 (BGP-4)*
RFC 2547	*BGP/MPLS VPNs*
RFC 2858	*Multiprotocol Extensions for BGP-4*

Technical Assistance

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

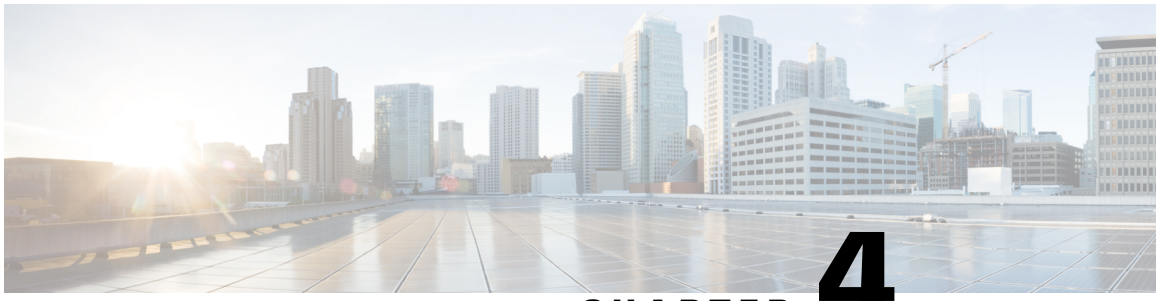
Feature Information for BGP NSF Awareness

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 NSF Awareness

Feature Name	Releases	Feature Information
BGP NSF Awareness	Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE Cisco IOS XE Release 3.6E	Nonstop Forwarding (NSF) awareness allows a device to assist NSF-capable neighbors to continue forwarding packets during a Stateful Switchover (SSO) operation. The BGP Nonstop Forwarding Awareness feature allows an NSF-aware device that is running BGP to forward packets along routes that are already known for a device that is performing an SSO operation. This capability allows the BGP peers of the failing device to retain the routing information that is advertised by the failing device and continue to use this information until the failed device has returned to normal operating behavior and is able to exchange routing information. The peering session is maintained throughout the entire NSF operation. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches. The following commands were introduced or modified: **bgp graceful-restart**, **show ip bgp**, **show ip bgp neighbors**. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.



CHAPTER 4

BGP Neighbor Policy

The BGP Neighbor Policy feature introduces new keywords to two existing commands to display information about local and inherited policies. When BGP neighbors use multiple levels of peer templates, it can be difficult to determine which policies are applied to the neighbor. Inherited policies are policies that the neighbor inherits from a peer group or a peer policy template.

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 Neighbor Policy

Benefit of BGP Neighbor Policy Feature

The BGP Neighbor Policy feature introduces new keywords to the **show ip bgp neighbors policy** command and the **show ip bgp template peer-policy** command to display information about local and inherited policies. When BGP neighbors use multiple levels of peer templates, it can be difficult to determine which policies are applied to the neighbor. Inherited policies are policies that the neighbor inherits from a peer group or a peer policy template.

How to Display BGP Neighbor Policy Information

Displaying BGP Neighbor Policy Information

SUMMARY STEPS

1. **enable**
2. **show ip bgp neighbors** { *ip-address* | *ipv6-address* } **policy** [**detail**]
3. **show ip bgp template peer-policy** [*policy-template-name* [**detail**]]

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 neighbors** { *ip-address* | *ipv6-address* } **policy** [**detail**] **Example:** `Device# show ip bgp neighbors 192.168.2.3 policy detail`	Displays the policies applied to the specified neighbor.
Step 3	**show ip bgp template peer-policy** [*policy-template-name* [**detail**]] **Example:** `Device# show ip bgp template peer-policy`	Displays the locally configured peer policy templates.

Additional References

Related Documents

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

Related Topic	Document Title
BGP commands	Cisco IOS IP Routing: BGP Command Reference

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

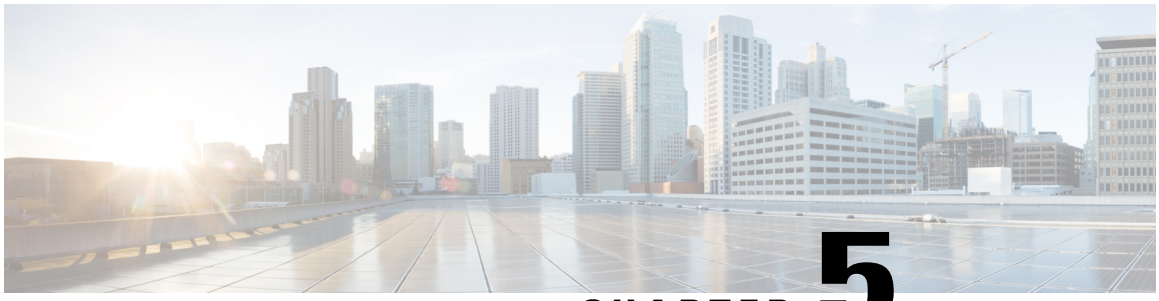
Feature Information for BGP Neighbor Policy

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 Neighbor Policy

Feature Name	Releases	Feature Information
BGP Neighbor Policy	Cisco IOS XE 3.1.0SG Cisco IOS XE Release 3.3SE Cisco IOS XE Release 3.6E	The BGP Neighbor Policy feature introduces new keywords to two existing commands to display information about local and inherited policies. When BGP neighbors use multiple levels of peer templates, it can be difficult to determine which policies are applied to the neighbor. Inherited policies are policies that the neighbor inherits from a peer-group or a peer-policy template. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches. The following commands were modified: **show ip bgp neighbors**, and **show ip bgp template peer-policy**. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.



CHAPTER 5

BGP Route-Map Continue

The BGP Route-Map Continue feature introduces the continue clause to BGP route-map configuration. The continue clause allows for more programmable policy configuration and route filtering and introduces the capability to execute additional entries in a route map after an entry is executed with successful match and set clauses. Continue clauses allow the network operator to configure and organize more modular policy definitions so that specific policy configuration need not be repeated within the same route map.

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 Route Map Continue

BGP Route Map with a Continue Clause

In BGP route-map configuration, the continue clause allows for more programmable policy configuration and route filtering and introduced the capability to execute additional entries in a route map after an entry is executed with successful match and set clauses. Continue clauses allow you to configure and organize more

modular policy definitions so that specific policy configurations need not be repeated within the same route map. Before the continue clause was introduced, route-map configuration was linear and did not allow any control over the flow of a route map.

Route Map Operation Without Continue Clauses

A route map evaluates match clauses until a successful match occurs. After the match occurs, the route map stops evaluating match clauses and starts executing set clauses, in the order in which they were configured. If a successful match does not occur, the route map "falls through" and evaluates the next sequence number of the route map until all configured route map entries have been evaluated or a successful match occurs. Each route map sequence is tagged with a sequence number to identify the entry. Route map entries are evaluated in order starting with the lowest sequence number and ending with the highest sequence number. If the route map contains only set clauses, the set clauses will be executed automatically, and the route map will not evaluate any other route map entries.

Route Map Operation with Continue Clauses

When a continue clause is configured, the route map will continue to evaluate and execute match clauses in the specified route map entry after a successful match occurs. The continue clause can be configured to go to (jump to) a specific route map entry by specifying the sequence number, or if a sequence number is not specified, the continue clause will go to the next sequence number. This behavior is called an "implied continue." If a match clause exists, the continue clause is executed only if a match occurs. If no successful matches occur, the continue clause is ignored.

Match Operations with Continue Clauses

If a match clause does not exist in the route map entry but a continue clause does, the continue clause will be automatically executed and go to the specified route map entry. If a match clause exists in a route map entry, the continue clause is executed only when a successful match occurs. When a successful match occurs and a continue clause exists, the route map executes the set clauses and then goes to the specified route map entry. If the next route map entry contains a continue clause, the route map will execute the continue clause if a successful match occurs. If a continue clause does not exist in the next route map entry, the route map will be evaluated normally. If a continue clause exists in the next route map entry but a match does not occur, the route map will not continue and will "fall through" to the next sequence number if one exists.

Note If the number of community lists in a match community clause within a route map exceed 256 characters in a line, you must nvgen multiple match community statements in a new line.

Set Operations with Continue Clauses

Set clauses are saved during the match clause evaluation process and are executed after the route-map evaluation is completed. The set clauses are evaluated and executed in the order in which they were configured. Set clauses are executed only after a successful match occurs, unless the route map does not contain a match clause. The continue statement proceeds to the specified route map entry only after configured set actions are performed. If a set action occurs in the first route map and then the same set action occurs again, with a different value, in a subsequent route map entry, the last set action may override any previous set actions that

were configured with the same **set** command unless the **set** command permits more than one value. For example, the **set as-path prepend** command permits more than one autonomous system number to be configured.

Note A continue clause can be executed, without a successful match, if a route map entry does not contain a match clause.

Note Route maps have a linear behavior, not a nested behavior. Once a route is matched in a route map permit entry with a continue command clause, it will not be processed by the implicit deny at the end of the route-map. For an example, see the "Examples: Filtering Traffic Using Continue Clauses in a BGP Route Map" section.

How to Filter Traffic Using Continue Clauses in a BGP Route Map

Filtering Traffic Using Continue Clauses in a BGP Route Map

Perform this task to filter traffic using continue clauses in a BGP route map.

Note Continue clauses can go only to a higher route map entry (a route map entry with a higher sequence number) and cannot go to a lower route map entry.

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*| *peer-group-name*} **route-map** *map-name* {**in** | **out**}
6. **exit**
7. **route-map** *map-name* {**permit** | **deny**} [*sequence-number*]
8. **match ip address** {*access-list-number* | *access-list-name*} [*... access-list-number* | *... access-list-name*]
9. **set community** *community-number* [**additive**] [*well-known-community*] | **none**}
10. **continue** [*sequence-number*]
11. **end**
12. **show route-map** [*map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 50000`	Enters router configuration mode, and creates a BGP routing process.
Step 4	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.0.0.1 remote-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.
Step 5	**neighbor** {*ip-address*| *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router)# neighbor 10.0.0.1 route-map ROUTE-MAP-NAME in`	Applies the inbound route map to routes received from the specified neighbor, or applies an outbound route map to routes advertised to the specified neighbor.
Step 6	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 7	**route-map** *map-name* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map ROUTE-MAP-NAME permit 10`	Enters route-map configuration mode to create or configure a route map.
Step 8	**match ip address** {*access-list-number* | *access-list-name*} [*... access-list-number* | *... access-list-name*]	Configures a **match** command that specifies the conditions under which policy routing and route filtering occur.

	Command or Action	Purpose
	Example: `Device(config-route-map)# match ip address 1`	• Multiple **match** commands can be configured. If a **match** command is configured, a match must occur in order for the continue statement to be executed. If a **match** command is not configured, set and continue clauses will be executed. **Note** The **match** and **set** commands used in this task are examples that are used to help describe the operation of the **continue** command. For a list of specific **match** and **set** commands, see the **continue** command in the *Cisco IOS IP Routing: BGP Command Reference*.
Step 9	**set community** *community-number* [**additive**] [*well-known-community*] | **none**} **Example:** `Device(config-route-map)# set community 10:1`	Configures a **set** command that specifies the routing action to perform if the criteria enforced by the **match** commands are met. • Multiple **set** commands can be configured. • In this example, a clause is created to set the specified community.
Step 10	**continue** [*sequence-number*] **Example:** `Device(config-route-map)# continue`	Configures a route map to continue to evaluate and execute match statements after a successful match occurs. • If a sequence number is configured, the continue clause will go to the route map with the specified sequence number. • If no sequence number is specified, the continue clause will go to the route map with the next sequence number. This behavior is called an "implied continue." **Note** Continue clauses in outbound route maps are supported in Cisco IOS XE Release 2.1 and later releases.
Step 11	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 12	**show route-map** [*map-name*] **Example:** `Device# show route-map`	(Optional) Displays locally configured route maps. The name of the route map can be specified in the syntax of this command to filter the output.

Examples

The following sample output shows how to verify the configuration of continue clauses using the **show route-map** command. The output displays configured route maps including the match, set, and continue clauses.

```
Device# show route-map

route-map MARKETING, permit, sequence 10
  Match clauses:
    ip address (access-lists): 1
    metric 10
  Continue: sequence 40
  Set clauses:
    as-path prepend 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 20
  Match clauses:
    ip address (access-lists): 2
    metric 20
  Set clauses:
    as-path prepend 10 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 30
  Match clauses:
  Continue: to next entry 40
  Set clauses:
    as-path prepend 10 10 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 40
  Match clauses:
    community (community-list filter): 10:1
  Set clauses:
    local-preference 104
  Policy routing matches: 0 packets, 0 bytes
route-map MKTG-POLICY-MAP, permit, sequence 10
  Match clauses:
  Set clauses:
    community 655370
  Policy routing matches: 0 packets, 0 bytes
```

Configuration Examples for BGP Route Map Continue

Examples: Filtering Traffic Using Continue Clauses in a BGP Route Map

The following example shows continue clause configuration in a route map sequence.

Note Continue clauses in outbound route maps are supported only in Cisco IOS Release 12.0(31)S, 12.2(33)SB, 12.2(33)SRB, 12.2(33)SXI, 12.4(4)T, and later releases.

The first continue clause in route map entry 10 indicates that the route map will go to route map entry 30 if a successful matches occurs. If a match does not occur, the route map will “fall through” to route map entry 20. If a successful match occurs in route map entry 20, the set action will be executed and the route map will not evaluate any additional route map entries. Only the first successful **match ip address** clause is supported.

If a successful match does not occur in route map entry 20, the route map will fall through to route map entry 30. This sequence does not contain a match clause, so the set clause will be automatically executed and the **continue** clause will go to the next route map entry because a sequence number is not specified.

If there are no successful matches, the route map will fall through to route map entry 30 and execute the set clause. A sequence number is not specified for the **continue** clause, so route map entry 40 will be evaluated.

There are two behaviors that can occur when the same **set** command is repeated in subsequent **continue** clause entries. For **set** commands that configure an additive or accumulative value (for example, **set community additive**, **set extended community additive**, and **set as-path prepend**), subsequent values are added by subsequent entries. The following example illustrates this behavior. After each set of match clauses, a **set as-path prepend** command is configured to add an autonomous system number to the as-path. After a match occurs, the route map stops evaluating match clauses and starts executing the set clauses, in the order in which they were configured. Depending on how many successful match clauses occur, the as-path is prepended by one, two, or three autonomous system numbers.

```
route-map ROUTE-MAP-NAME permit 10
 match ip address 1
 match metric 10
 set as-path prepend 10
 continue 30
 !
route-map ROUTE-MAP-NAME permit 20
 match ip address 2
 match metric 20
 set as-path prepend 10 10
 !
route-map ROUTE-MAP-NAME permit 30
 set as-path prepend 10 10 10
 continue
 !
route-map ROUTE-MAP-NAME permit 40
 match community 10:1
 set local-preference 104
```

In this example, the same **set** command is repeated in subsequent **continue** clause entries, but the behavior is different from the first example. For **set** commands that configure an absolute value, the value from the last instance will overwrite the previous value(s). The following example illustrates this behavior. The set clause value in sequence 20 overwrites the set clause value from sequence 10. The next hop for prefixes from the 172.16/16 network is set to 10.2.2.2, not 10.1.1.1.

```
ip prefix-list 1 permit 172.16.0.0/16
ip prefix-list 2 permit 192.168.1.0/24
route-map RED permit 10
 match ip address prefix-list 1
 set ip next hop 10.1.1.1
 continue 20
 exit
route-map RED permit 20
 match ip address prefix-list 2
 set ip next hop 10.2.2.2
 end
```

Note Route maps have a linear behavior and not a nested behavior. Once a route is matched in a route map permit entry with a continue command clause, it will not be processed by the implicit deny at the end of the route-map. The following example illustrates this case.

In the following example, when routes match an as-path of 10, 20, or 30, the routes are permitted and the continue clause jumps over the explicit deny clause to process the match ip address prefix list. If a match occurs here, the route metric is set to 100. Only routes that do not match an as-path of 10, 20, or 30 and do match a community number of 30 are denied. To deny other routes, you must configure an explicit deny statement.

```
route-map test permit 10
 match as-path 10 20 30
```

```
 continue 30
 exit
route-map test deny 20
 match community 30
 exit
route-map test permit 30
 match ip address prefix-list 1
 set metric 100
 exit
```

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

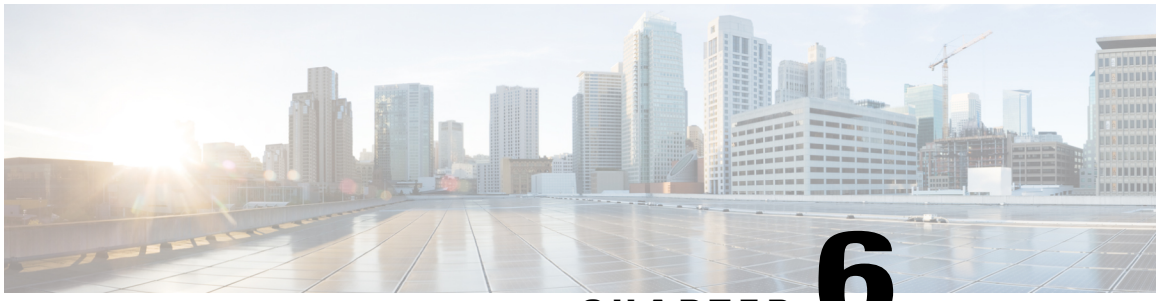
Feature Information for BGP Route Map Continue

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 Route Map Continue

Feature Name	Releases	Feature Information
BGP Route Map Continue	Cisco IOS XE 3.1.0SG Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE Cisco IOS XE Release 3.6E	The BGP Route Map Continue feature introduces the continue clause to BGP route map configuration. The continue clause allows for more programmable policy configuration and route filtering and introduces the capability to execute additional entries in a route map after an entry is executed with successful match and set clauses. Continue clauses allow the network operator to configure and organize more modular policy definitions so that specific policy configuration need not be repeated within the same route map. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches. In Cisco IOS XE Release 3.6E, this feature is supported on Cisco Catalyst 3850 Series Switches.



CHAPTER 6

BGP Route-Map Continue Support for Outbound Policy

The BGP Route-Map Continue Support for an Outbound Policy feature introduces support for continue clauses to be applied to outbound route maps.

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.

Iinformation About BGP Route-Map Continue Support for Outbound Policy

BGP Route Map with a Continue Clause

Subsequent to the Cisco implementation of route maps, the continue clause was introduced into BGP route map configuration. The continue clause allows for more programmable policy configuration and route filtering. The continue clause introduces the ability to execute additional entries in a route map after an entry is executed with successful match and set clauses. Continue clauses allow you to configure and organize more modular policy definitions so that specific policy configurations need not be repeated within the same route map. Before the continue clause was introduced, route map configuration was linear and did not allow any control over the flow of a route map.

Route Map Operation Without Continue Clauses

A route map evaluates match clauses until a successful match occurs. After the match occurs, the route map stops evaluating match clauses and starts executing set clauses, in the order in which they were configured. If a successful match does not occur, the route map "falls through" and evaluates the next sequence number of the route map until all configured route map entries have been evaluated or a successful match occurs. Each route map sequence is tagged with a sequence number to identify the entry. Route map entries are evaluated in order starting with the lowest sequence number and ending with the highest sequence number. If the route map contains only set clauses, the set clauses will be executed automatically, and the route map will not evaluate any other route map entries.

Route Map Operation with Continue Clauses

When a continue clause is configured, the route map will continue to evaluate and execute match clauses in the specified route map entry after a successful match occurs. The continue clause can be configured to go to (jump to) a specific route map entry by specifying the sequence number, or if a sequence number is not specified, the continue clause will go to the next sequence number. This behavior is called an "implied continue." If a match clause exists, the continue clause is executed only if a match occurs. If no successful matches occur, the continue clause is ignored.

Match Operations with Continue Clauses

If a match clause does not exist in the route map entry but a continue clause does, the continue clause will be automatically executed and go to the specified route map entry. If a match clause exists in a route map entry, the continue clause is executed only when a successful match occurs. When a successful match occurs and a continue clause exists, the route map executes the set clauses and then goes to the specified route map entry. If the next route map entry contains a continue clause, the route map will execute the continue clause if a successful match occurs. If a continue clause does not exist in the next route map entry, the route map will be evaluated normally. If a continue clause exists in the next route map entry but a match does not occur, the route map will not continue and will "fall through" to the next sequence number if one exists.

Note If the number of community lists in a match community clause within a route map exceed 256 characters in a line, you must nvgen multiple match community statements in a new line.

Set Operations with Continue Clauses

Set clauses are saved during the match clause evaluation process and are executed after the route-map evaluation is completed. The set clauses are evaluated and executed in the order in which they were configured. Set clauses are executed only after a successful match occurs, unless the route map does not contain a match clause. The continue statement proceeds to the specified route map entry only after configured set actions are performed. If a set action occurs in the first route map and then the same set action occurs again, with a different value, in a subsequent route map entry, the last set action may override any previous set actions that were configured with the same **set** command unless the **set** command permits more than one value. For example, the **set as-path prepend** command permits more than one autonomous system number to be configured.

Note A continue clause can be executed, without a successful match, if a route map entry does not contain a match clause.

Note Route maps have a linear behavior, not a nested behavior. Once a route is matched in a route map permit entry with a continue command clause, it will not be processed by the implicit deny at the end of the route-map. For an example, see the “Examples: Filtering Traffic Using Continue Clauses in a BGP Route Map” section.

How to Filter Traffic Using Continue Clauses in a BGP Route Map

Filtering Traffic Using Continue Clauses in a BGP Route Map

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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
7. **exit**
8. **exit**
9. **route-map** *map-name* {**permit** | **deny**} [*sequence-number*]
10. **match ip address** {*access-list-number* | *access-list-name*} [*... access-list-number* | *... access-list-name*]
11. **set community** { { [*community-number*] [*well-known-community*] [**additive**]} | **none**}
12. **continue** [*sequence-number*]
13. **end**
14. **show route-map** [*map-name*]

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 50000`	Enters router configuration mode, and creates a BGP routing process.

	Command or Action	Purpose
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.0.0.1 remote-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 device.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in 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 6	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 10.0.0.1 route-map ROUTE-MAP-NAME in`	Applies the inbound route map to routes received from the specified neighbor, or applies an outbound route map to routes advertised to the specified neighbor.
Step 7	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 8	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 9	**route-map** *map-name* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map ROUTE-MAP-NAME permit 10`	Enters route-map configuration mode to create or configure a route map.

	Command or Action	Purpose
Step 10	**match ip address** {*access-list-number* | *access-list-name*} [*... access-list-number* | *... access-list-name*] **Example:** `Device(config-route-map)# match ip address 1`	Configures a **match** command that specifies the conditions under which policy routing and route filtering occur. • Multiple **match** commands can be configured. If a **match** command is configured, a match must occur in order for the continue statement to be executed. If a **match** command is not configured, set and continue clauses will be executed. **Note** The **match** and **set** commands used in this task are examples that are used to help describe the operation of the **continue** command. For a list of specific **match** and **set** commands, see the **continue** command in the *Cisco IOS IP Routing: BGP Command Reference*.
Step 11	**set community** { { [*community-number*] [*well-known-community*] [**additive**]} | **none**} **Example:** `Device(config-route-map)# set community 10:1`	Configures a **set** command that specifies the routing action to perform if the criteria enforced by the **match** commands are met. • Multiple **set** commands can be configured. • In this example, a clause is created to set the specified community number in aa:nn format.
Step 12	**continue** [*sequence-number*] **Example:** `Device(config-route-map)# continue`	Configures a route map to continue to evaluate and execute match statements after a successful match occurs. • If a sequence number is configured, the continue clause will go to the route map with the specified sequence number. • If no sequence number is specified, the continue clause will go to the route map with the next sequence number. This behavior is called an “implied continue.”
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show route-map** [*map-name*] **Example:** `Device# show route-map`	(Optional) Displays locally configured route maps. The name of the route map can be specified in the syntax of this command to filter the output.

Examples

The following sample output shows how to verify the configuration of continue clauses using the **show route-map** command. The output displays configured route maps including the match, set, and continue clauses.

```
Device# show route-map

route-map MARKETING, permit, sequence 10
  Match clauses:
    ip address (access-lists): 1
    metric 10
  Continue: sequence 40
  Set clauses:
    as-path prepend 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 20
  Match clauses:
    ip address (access-lists): 2
    metric 20
  Set clauses:
    as-path prepend 10 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 30
  Match clauses:
  Continue: to next entry 40
  Set clauses:
    as-path prepend 10 10 10
  Policy routing matches: 0 packets, 0 bytes
route-map MARKETING, permit, sequence 40
  Match clauses:
    community (community-list filter): 10:1
  Set clauses:
    local-preference 104
  Policy routing matches: 0 packets, 0 bytes
route-map MKTG-POLICY-MAP, permit, sequence 10
  Match clauses:
  Set clauses:
    community 655370
  Policy routing matches: 0 packets, 0 bytes
```

Configuration Examples for BGP Route-Map Continue Support for Outbound Policy

Examples: Filtering Traffic Using Continue Clauses in a BGP Route Map

The following example shows continue clause configuration in a route map sequence.

The first continue clause in route map entry 10 indicates that the route map will go to route map entry 30 if a successful matches occurs. If a match does not occur, the route map will “fall through” to route map entry 20. If a successful match occurs in route map entry 20, the set action will be executed and the route map will not evaluate any additional route map entries. Only the first successful match ip address clause is supported.

If a successful match does not occur in route map entry 20, the route map will fall through to route map entry 30. This sequence does not contain a match clause, so the set clause will be automatically executed and the continue clause will go to the next route map entry because a sequence number is not specified.

If there are no successful matches, the route map will fall through to route map entry 30 and execute the set clause. A sequence number is not specified for the continue clause, so route map entry 40 will be evaluated.

There are two behaviors that can occur when the same **set** command is repeated in subsequent continue clause entries. For **set** commands that configure an additive or accumulative value (for example, **set community additive**, **set extended community additive**, and **set as-path prepend**), subsequent values are added by subsequent entries. The following example illustrates this behavior. After each set of match clauses, a **set as-path prepend** command is configured to add an autonomous system number to the as-path. After a match occurs, the route map stops evaluating match clauses and starts executing the set clauses, in the order in which they were configured. Depending on the number of successful match clauses, the as-path is prepended by one, two, or three autonomous system numbers.

```
route-map ROUTE-MAP-NAME permit 10
 match ip address 1
 match metric 10
 set as-path prepend 10
 continue 30
 !
route-map ROUTE-MAP-NAME permit 20
 match ip address 2
 match metric 20
 set as-path prepend 10 10
 !
route-map ROUTE-MAP-NAME permit 30
 set as-path prepend 10 10 10
 continue
 !
route-map ROUTE-MAP-NAME permit 40
 match community 10:1
 set local-preference 104
```

In this example, the same **set** command is repeated in subsequent continue clause entries but the behavior is different from the first example. For **set** commands that configure an absolute value, the value from the last instance will overwrite the previous value(s). The following example illustrates this behavior. The set clause value in sequence 20 overwrites the set clause value from sequence 10. The next hop for prefixes from the 172.16/16 network is set to 10.2.2.2 and not 10.1.1.1.

```
ip prefix-list 1 permit 172.16.0.0/16
ip prefix-list 2 permit 192.168.1.0/24
route-map RED permit 10
 match ip address prefix-list 1
 set ip next hop 10.1.1.1
 continue 20
 exit
route-map RED permit 20
 match ip address prefix-list 2
 set ip next hop 10.2.2.2
 end
```

Note Route maps have a linear behavior, not a nested behavior. Once a route is matched in a route map permit entry with a continue command clause, it will not be processed by the implicit deny at the end of the route map. The following example illustrates this case.

In the following example, when routes match an AS-path of 10, 20, or 30, the routes are permitted and the continue clause jumps over the explicit deny clause to process the **match ip address prefix-list** command. If a match occurs here, the route metric is set to 100. Only routes that do not match an AS-path of 10, 20, or 30 and do match a community number of 30 are denied. To deny other routes, you must configure an explicit deny statement.

```
route-map test permit 10
 match as-path 10 20 30
 continue 30
 exit
route-map test deny 20
```

```
 match community 30
 exit
route-map test permit 30
 match ip address prefix-list 1
 set metric 100
 exit
```

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

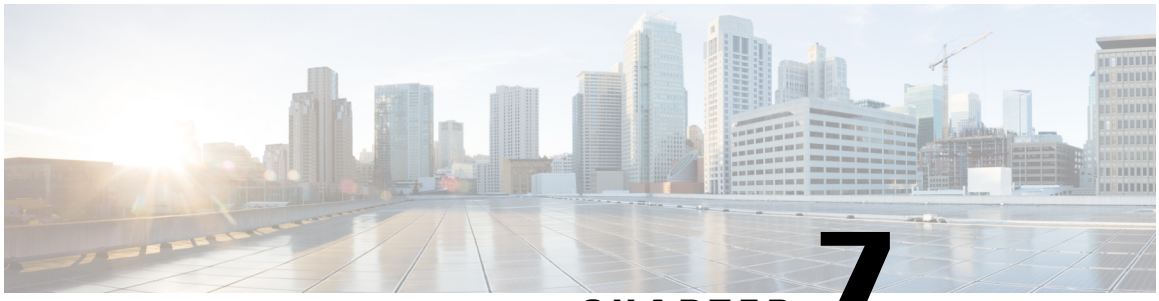
Feature Information for BGP Route-Map Continue Support for Outbound Policy

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 Route-Map Continue Support for Outbound Policy

Feature Name	Releases	Feature Information
BGP Route-Map Continue Support for Outbound Policy	Cisco IOS XE Release 3.1.0SG Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE	The BGP Route-Map Continue Support for an Outbound Policy feature introduces support for continue clauses to be applied to outbound route maps. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 7

IPv6 Routing: Multiprotocol BGP Extensions for IPv6

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 IPv6 Routing: Multiprotocol BGP Extensions for IPv6

Multiprotocol BGP Extensions for IPv6

Multiprotocol BGP is the supported Exterior Gateway Protocol (EGP) for IPv6. Multiprotocol BGP extensions for IPv6 supports many of the same features and functionality as IPv4 BGP. IPv6 enhancements to multiprotocol BGP include support for an IPv6 address family and Network Layer Reachability Information (NLRI) and next hop (the next device in the path to the destination) attributes that use IPv6 addresses.

How to Implement Multiprotocol BGP for IPv6

Configuring an IPv6 BGP Routing Process and BGP Router ID

Perform this task to configure an IPv6 BGP routing process and an optional BGP router ID for a BGP-speaking device.

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

When configuring BGP on a device that is enabled only for IPv6 (that is, the device does not have an IPv4 address), you must manually configure the BGP router ID for the device. The BGP router ID, which is represented as a 32-bit value using an IPv4 address syntax, must be unique to the BGP peers of the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **bgp router-id** *ip-address*

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 for the specified routing process.

	Command or Action	Purpose
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process specified in the previous step. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** command unless you configure the **no bgp default ipv4-unicast** command before configuring the **neighbor remote-as** command.
Step 5	**bgp router-id** *ip-address* **Example:** `Device(config-router)# bgp router-id 192.168.99.70`	(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP. **Note** Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.

Configuring IPv6 Multiprotocol BGP Between Two Peers

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, as shown for IPv6 prefixes.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address* [**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number ...*]
5. **address-family ipv6** [**unicast** | **multicast**]
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**

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`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* [**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 2001:DB8:0:CC00::1 remote-as 64600`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device.
Step 5	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device.

Advertising Routes into IPv6 Multiprotocol BGP

By default, networks that are defined in router configuration mode using the **network** command are injected into the IPv4 unicast database. To inject a network into another database, such as the IPv6 BGP database, you must define the network using the **network** command in address family configuration mode for the other database, as shown for the IPv6 BGP database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **network** {*network-number* [**mask** *network-mask*] | *nsap-prefix*} [**route-map** *map-tag*]
6. **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	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast**	**multicast**	**vpnv6**] **Example:** `Device(config-router)# address-family ipv6 unicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**network** {*network-number* [**mask** *network-mask*]	*nsap-prefix*} [**route-map** *map-tag*] **Example:** `Device(config-router-af)# network 2001:DB8::/24`	Advertises (injects) the specified prefix into the IPv6 BGP database (the routes must first be found in the IPv6 unicast routing table). • The prefix is injected into the database for the address family specified in the previous step. • Routes are tagged from the specified prefix as “local origin.” • The *ipv6-prefix* argument in the **network** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.

	Command or Action	Purpose
		• The *prefix-length* argument is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash mark must precede the decimal value.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode. • Repeat this step to exit router configuration mode and return the device to global configuration mode.

Configuring a Route Map for IPv6 Multiprotocol BGP Prefixes

- 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, as shown for IPv6 prefixes.
- By default, route maps that are applied in router configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. Route maps for other address families must be applied in address family configuration mode using the **neighbor route-map** command, as shown for the IPv6 address family. The route maps are applied either as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number ...*]
5. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
6. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}
8. **exit**
9. **exit**
10. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
11. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}

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`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 2001:DB8:0:cc00::1 remote-as 64600`	Adds the link-local IPv6 address of the neighbor in the specified remote autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device.
Step 5	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 6	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:cc00::1 activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device using the specified link-local addresses.
Step 7	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}	Applies a route map to incoming or outgoing routes. • Changes to the route map will not take effect for existing peers until the peering is reset or a soft reset is performed.

	Command or Action	Purpose
	Example: `Device(config-router-af)# neighbor 2001:DB8:0:cc00::1 route-map rtp in`	Using the **clear bgp ipv6** command with the **soft** and **in** keywords will perform a soft reset.
Step 8	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.
Step 9	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 10	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map rtp permit 10`	Defines a route map and enters route-map configuration mode. • Follow this step with a **match** command.
Step 11	**match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*} **Example:** `Device(config-route-map)# match ipv6 address prefix-list cisco`	Distributes any routes that have a destination IPv6 network number address permitted by a prefix list, or performs policy routing on packets.

Redistributing Prefixes into IPv6 Multiprotocol BGP

Redistribution is the process of redistributing, or injecting, prefixes from one routing protocol into another routing protocol. This task explains how to inject prefixes from a routing protocol into IPv6 multiprotocol BGP. Specifically, prefixes that are redistributed into IPv6 multiprotocol BGP using the **redistribute** router configuration command are injected into the IPv6 unicast database.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*]
6. **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	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified with the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 5	**redistribute bgp** [*process-id*] [**metric** *metric-value*] [**route-map** *map-name*] [*source-protocol-options*] **Example:** `Device(config-router-af)# redistribute bgp 64500 metric 5`	Redistributes IPv6 routes from one routing domain into another routing domain.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode. • Repeat this step to exit router configuration mode and return the device to global configuration mode.

Clearing External BGP Peers

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **external** [**soft**] [**in** | **out**]
3. **clear bgp ipv6** {**unicast** | **multicast**} **peer-group** *name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear bgp ipv6** {**unicast**	**multicast**} **external** [**soft**] [**in**	**out**] **Example:** `Device# clear bgp ipv6 unicast external soft in`	Clears external IPv6 BGP peers.
Step 3	**clear bgp ipv6** {**unicast**	**multicast**} **peer-group** *name* **Example:** `Device# clear bgp ipv6 unicast peer-group marketing`	Clears all members of an IPv6 BGP peer group.

Advertising IPv4 Routes Between IPv6 BGP Peers

If an IPv6 network is connecting two separate IPv4 networks, IPv6 can be used to advertise the IPv4 routes. Configure the peering using the IPv6 addresses within the IPv4 address family. Set the next hop with a static route or with an inbound route map because the advertised next hop will usually be unreachable. Advertising IPv6 routes between two IPv4 peers is also possible using the same model.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** *peer-group-name* **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...]
6. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
7. **neighbor** *ipv6-address* **peer-group** *peer-group-name*
8. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}
9. **exit**
10. **exit**
11. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address* [... *ip-address*] [**peer-address**]

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`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor 6peers peer-group`	Creates a multiprotocol BGP peer group.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number* ...] **Example:** `Device(config-router)# neighbor 6peers remote-as 65002`	Adds the IPv6 address of the neighbor in the specified autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device.
Step 6	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode to configure a routing session using standard IPv4 address prefixes.
Step 7	**neighbor** *ipv6-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 2001:DB8:1234::2 peer-group 6peers`	Assigns the IPv6 address of a BGP neighbor to a peer group.
Step 8	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 6peers route-map rmap out`	Applies a route map to incoming or outgoing routes. • Changes to the route map will not take effect for existing peers until the peering is reset or a soft reset is performed. Using the **clear bgp ipv6** command with the **soft** and **in** keywords will perform a soft reset.
Step 9	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 11	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map rmap permit 10`	Defines a route map and enters route-map configuration mode.
Step 12	**set ip next-hop** *ip-address* [... *ip-address*] [**peer-address**] **Example:** `Device(config-route-map)# set ip next-hop 10.21.8.10`	Overrides the next hop advertised to the peer for IPv4 packets.

Configuration Examples for Multiprotocol BGP for IPv6

Example: Configuring a BGP Process, BGP Router ID, and IPv6 Multiprotocol BGP Peer

The following example enables IPv6 globally, configures a BGP process, and establishes a BGP router ID. Also, the IPv6 multiprotocol BGP peer 2001:DB8:0:CC00::1 is configured and activated.

```
ipv6 unicast-routing
!
router bgp 65000
 no bgp default ipv4-unicast
 bgp router-id 192.168.99.70
 neighbor 2001:DB8:0:CC00::1 remote-as 64600
 address-family ipv6 unicast
  neighbor 2001:DB8:0:CC00::1 activate
```

Example: Configuring an IPv6 Multiprotocol BGP Peer Group

The following example configures the IPv6 multiprotocol BGP peer group named group1:

```
router bgp 65000
no bgp default ipv4-unicast
neighbor group1 peer-group
neighbor 2001:DB8:0:CC00::1 remote-as 64600
address-family ipv6 unicast
 neighbor group1 activate
 neighbor 2001:DB8:0:CC00::1 peer-group group1
```

Example: Advertising Routes into IPv6 Multiprotocol BGP

The following example injects the IPv6 network 2001:DB8::/24 into the IPv6 unicast database of the local device. (BGP checks that a route for the network exists in the IPv6 unicast database of the local device before advertising the network.)

```
router bgp 65000
 no bgp default ipv4-unicast
address-family ipv6 unicast
  network 2001:DB8::/24
```

Example: Configuring a Route Map for IPv6 Multiprotocol BGP Prefixes

The following example configures the route map named rtp to permit IPv6 unicast routes from network 2001:DB8::/24 if they match the prefix list named cisco:

```
router bgp 64900
no bgp default ipv4-unicast
```

```
neighbor 2001:DB8:0:CC00::1 remote-as 64700
address-family ipv6 unicast
 neighbor 2001:DB8:0:CC00::1 activate
 neighbor 2001:DB8:0:CC00::1 route-map rtp in
ipv6 prefix-list cisco seq 10 permit 2001:DB8::/24
route-map rtp permit 10
 match ipv6 address prefix-list cisco
```

Example: Redistributing Prefixes into IPv6 Multiprotocol BGP

The following example redistributes RIP routes into the IPv6 unicast database of the local device:

```
router bgp 64900
no bgp default ipv4-unicast
address-family ipv6 unicast
 redistribute rip
```

Example: Advertising IPv4 Routes Between IPv6 Peers

The following example advertises IPv4 routes between IPv6 peers when the IPv6 network is connecting two separate IPv4 networks. Peering is configured using IPv6 addresses in the IPv4 address family configuration mode. The inbound route map named rmap sets the next hop because the advertised next hop is likely to be unreachable.

```
router bgp 65000
!
 neighbor 6peers peer-group
 neighbor 2001:DB8:1234::2 remote-as 65002
 address-family ipv4
 neighbor 6peers activate
 neighbor 6peers soft-reconfiguration inbound
 neighbor 2001:DB8:1234::2 peer-group 6peers
 neighbor 2001:DB8:1234::2 route-map rmap in
!
route-map rmap permit 10
 set ip next-hop 10.21.8.10
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Cisco IOS Master Command List, All Releases
IPv6 commands	Cisco IOS IPv6 Command Reference
Cisco IOS IPv6 features	Cisco IOS IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	IPv6 RFCs

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

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

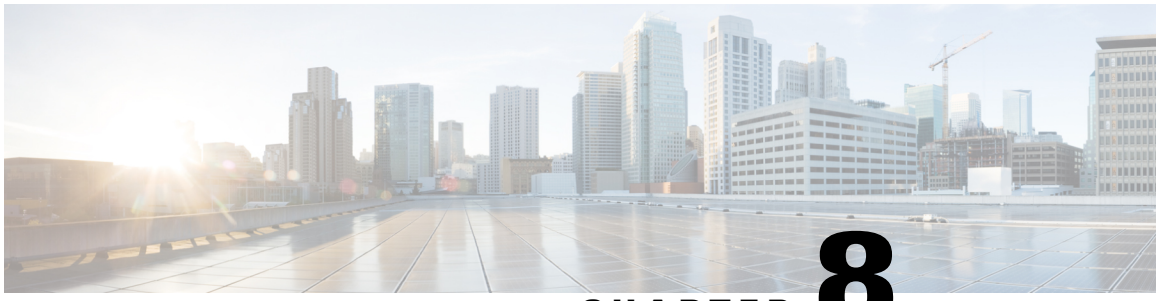
Feature Information for IPv6 Routing: Multiprotocol BGP Extensions for IPv6

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 IPv6 Routing: Multiprotocol BGP Extensions for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing: Multiprotocol BGP Extensions for IPv6	Cisco IOS XE Release 3.3SE	Multiprotocol BGP Extensions for IPv6 supports the same features and functionality as IPv4 BGP. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 8

IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

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 IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

IPv6 Multiprotocol BGP Peering Using a Link-Local Address

The IPv6 multiprotocol BGP can be configured between two IPv6 devices (peers) using link-local addresses. For this function to work, you must identify the interface for the neighbor by using the **neighbor update-source** command, and you must configure a route map to set an IPv6 global next hop.

Boarder Gateway Protocol (BGP) uses third-party next hops for peering with multiple peers over IPv6 link-local addresses on the same interface. Peering over link-local addresses on different interfaces cannot use third party next hops. The neighbors peering using link-local addresses are split into one update group per interface. BGP splits update group membership for neighbors with link-local addresses based on the interface used to communicate with that neighbor.

How to Configure IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

Configuring an IPv6 Multiprotocol BGP Peer Using a Link-Local Address

Configuring IPv6 multiprotocol BGP between two IPv6 devices (peers) using link-local addresses requires that you identify the interface for the neighbor by using the **neighbor update-source** command and that a you configure a route map to set an IPv6 global next hop.

Note

- By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To be able 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, as shown for IPv6 prefixes.
- By default, route maps that are applied in router configuration mode using the **neighbor route-map** command are applied to only IPv4 unicast address prefixes. Route maps for other address families must be applied in address family configuration mode using the **neighbor route-map** command, as shown for the IPv6 address family. The route maps are applied either as the inbound or outbound routing policy for neighbors under the specified address family. Configuring separate route maps under each address family type simplifies managing complicated or different policies for each address family.
- The route-map used to modify the next hop needs to be applied outbound only. Inbound route-map to modify next-hop ipv6 address is not supported. Inbound route-map is supported only for IPV4 address family.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address* [**%**] **peer-group**}
6. **neighbor** {*ip-address* | *ipv6-address %*| *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]
7. **neighbor** {*ip-address* | *ipv6-address* [**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number ...*]
8. **neighbor** {*ip-address* | *ipv6-address* [**%**] | *peer-group-name*} **update-source** *interface-type interface-number*
9. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
10. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
11. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**}
12. **exit**
13. **exit**
14. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
15. **match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*}
16. **set ipv6 next-hop** *ipv6-address* [*link-local-address*] [**peer-address**]
17. **exit**
18. **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** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address*	*ipv6-address*	*peer-group-name*} **peer-group**	Creates a BGP or multiprotocol BGP peer group.

	Command or Action	Purpose
	Example: `Device(config-router)# neighbor internal peer-group`	
Step 5	**neighbor** {*ip-address* | *ipv6-address* [**%**] **peer-group**} **Example:** `Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% peer-group`	Configures a BGP neighbor to member of a peer group. **Note** **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address %*| *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `Device(config-router)# neighbor internal remote-as 100`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 7	**neighbor** {*ip-address* | *ipv6-address* [**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number ...*] **Example:** `Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% remote-as 64600`	Adds the link-local IPv6 address of the neighbor in the specified remote autonomous system to the IPv6 multiprotocol BGP neighbor table of the local device. **Note** **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 8	**neighbor** {*ip-address* | *ipv6-address* [**%**] | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% update-source Gigabitethernet 0/0`	Specifies the link-local address over which the peering is to occur. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface. • If there are multiple connections to the neighbor and you do not specify the neighbor interface by using the *interface-type* and *interface-number* arguments in the **neighbor update-source** command, a TCP connection cannot be established with the neighbor using link-local addresses.
Step 9	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the device is placed in configuration mode for the IPv6 unicast address family if the **unicast** keyword is not specified with the **address-family ipv6** command.

	Command or Action	Purpose
		• The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 10	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Device(config-router-af)# neighbor FE80::1234:BFF:FE0E:A471% activate`	Enables the neighbor to exchange prefixes for the IPv6 address family with the local device using the specified link-local addresses. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 11	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* [**%**]} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor FE80::1234:BFF:FE0E:A471% route-map nh6 out`	Applies a route map to incoming or outgoing routes. • The optional **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 12	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode, and returns the device to router configuration mode.
Step 13	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode, and returns the device to global configuration mode.
Step 14	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map nh6 permit 10`	Defines a route map and enters route-map configuration mode.
Step 15	**match ipv6 address** {**prefix-list** *prefix-list-name* | *access-list-name*} **Example:** `Device(config-route-map)# match ipv6 address prefix-list cisco`	Distributes any routes that have a destination IPv6 network number address permitted by a prefix list, or performs policy routing on packets.
Step 16	**set ipv6 next-hop** *ipv6-address* [*link-local-address*] [**peer-address**] **Example:** `Device(config-route-map)# set ipv6 next-hop 2001:DB8::1`	Overrides the next hop advertised to the peer for IPv6 packets that pass a match clause of a route map for policy routing. • The *ipv6-address* argument specifies the IPv6 global address of the next hop. It need not be an adjacent device. • The *link-local-address* argument specifies the IPv6 link-local address of the next hop. It must be an adjacent device. If you do not specify this optional argument, the link-local address of

	Command or Action	Purpose
		the interface specified with the *interface-type* argument (in the **neighbor update-source** command in Step 5) is included as the next-hop in the BGP updates. Therefore, only one route map that sets the global IPv6 next-hop address in BGP updates is required for multiple BGP peers that use link-local addresses. • The route map sets the IPv6 next-hop addresses (global and link-local) in BGP updates. If the route map is not configured, the next-hop address in the BGP updates defaults to the unspecified IPv6 address (::), which is rejected by the peer.
Step 17	**exit** **Example:** `Device(config-router-map)# exit`	Exits router map configuration mode, and returns the device to router configuration mode.
Step 18	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode, and enters privileged EXEC mode.

Configuration Examples for IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

Example: Configuring an IPv6 Multiprotocol BGP Peer Using a Link-Local Address

The following example configures the IPv6 multiprotocol BGP peer FE80::1234:BFF:FE0E:A471 over Gigabitethernet interface 0/0 and sets the route map named nh6 to include the IPv6 next-hop global address of Gigabitethernet interface 0/0 in BGP updates. The IPv6 next-hop link-local address can be set by the nh6 route map (not shown in the following example) or from the interface specified by the **neighbor update-source** command (as shown in this example).

```
Device> enable
Device# configure terminal
Device(config)# router bgp 5
Device(config-router)# neighbor internal peer-group
Device(config-router)# neighbor  FE80::1234:BFF:FE0E:A471% peer-group
Device(config-router)# neighbor internal remote-as 100
Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% remote-as 64600
Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% update-source Gigabitethernet 0/0

Device(config-router)# address-family ipv6
Device(config-router-af)# neighbor FE80::1234:BFF:FE0E:A471% activate
Device(config-router-af)# neighbor FE80::1234:BFF:FE0E:A471% route-map nh6 out
```

```
Device(config-router-af)# exit
Device(config-router)# exit
Device(config)# route-map nh6permit 10
Device(config-router-map)# match ipv6 address prefix-list cisco
Device(config-router-map)# set ipv6 next-hop 2001:DB8:526::1
Device(config-router-map)# exit
Device(config)# ipv6 prefix-list cisco permit 2001:DB8:2F22::/48 le 128
Device(config)# ipv6 prefix-list cisco deny ::/0
Device(config)# end
```

Note If you specify only the global IPv6 next-hop address (the *ipv6-address* argument) with the **set ipv6 next-hop** command after specifying the neighbor interface (the *interface-type* argument) with the **neighbor update-source** command, the link-local address of the interface specified with the *interface-type* argument is included as the next hop in the BGP updates. Therefore, only one route map that sets the global IPv6 next-hop address in BGP updates is required for multiple BGP peers that use link-local addresses.

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Cisco IOS Master Command List, All Releases
IPv6 commands	Cisco IOS IPv6 Command Reference
Cisco IOS IPv6 features	Cisco IOS IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	IPv6 RFCs

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

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

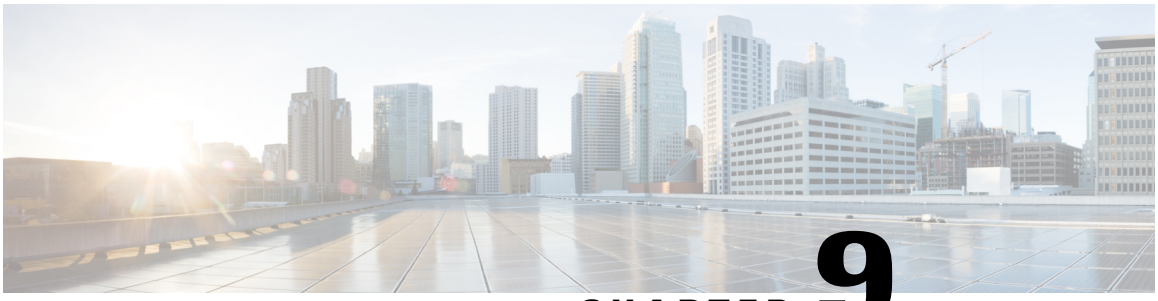
Feature Information for IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

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 IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

Feature Name	Releases	Feature Information
IPv6 Routing: Multiprotocol BGP Link-Local Address Peering	Cisco IOS XE 3.3SG Cisco IOS XE Release 3.3SE	IPv6 supports multiprotocol BGP link-local address peering. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 9

BGP Restart Neighbor Session After Max-Prefix Limit Reached

The BGP Restart Session After Max-Prefix Limit Reached feature adds the **restart** keyword to the **neighbor maximum-prefix** command. This allows a network operator to configure the time interval at which a peering session is reestablished by a device when the number of prefixes that have been received from a peer has exceeded the maximum prefix limit.

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 Neighbor Session Restart After Max-Prefix Limit Reached

Prefix Limits and BGP Peering Sessions

Use the **neighbor maximum-prefix** command to limit the maximum number of prefixes that a device running BGP can receive from a peer. When the device receives too many prefixes from a peer and the maximum-prefix limit is exceeded, the peering session is disabled or brought down. The session stays down until the network operator manually brings the session back up by entering the **clear ip bgp** command, which clears stored prefixes.

BGP Neighbor Session Restart with the Maximum Prefix Limit

The **restart** keyword was added to the **neighbor maximum-prefix** command so that a network operator can configure a device to automatically reestablish a BGP neighbor peering session when the peering session has been disabled or brought down. The time interval at which peering can be reestablished automatically is configurable. The *restart-interval* for the **restart** keyword is specified in minutes; range is from 1 to 65,535 minutes.

Subcodes for BGP Cease Notification

Border Gateway Protocol (BGP) imposes maximum limits on the maximum number of prefixes that are accepted from a peer for a given address family. This limitation safeguards the device from resource depletion caused by misconfiguration, either locally or on the remote neighbor. To prevent a peer from flooding BGP with advertisements, a limit is placed on the number of prefixes that are accepted from a peer for each supported address family. The default limits can be overridden through configuration of the maximum-prefix limit command for the peer for the appropriate address family.

The following subcodes are supported for the BGP cease notification message:

- Maximum number of prefixes reached
- Administrative shutdown
- Peer de-configured
- Administrative reset

A cease notification message is sent to the neighbor and the peering with the neighbor is terminated when the number of prefixes received from the peer for a given address family exceeds the maximum limit (either set by default or configured by the user) for that address family. It is possible that the maximum number of prefixes for a neighbor for a given address family has been configured after the peering with the neighbor has been established and a certain number of prefixes have already been received from the neighbor for that address family. A cease notification message is sent to the neighbor and peering with the neighbor is terminated immediately after the configuration if the configured maximum number of prefixes is fewer than the number of prefixes that have already been received from the neighbor for the address family.

How to Configure a Device to Reestablish a Neighbor Session After the Maximum Prefix Limit Has Been Exceeded

Configuring a Router to Reestablish a Neighbor Session After the Maximum Prefix Limit Reached

Perform this task to configure the time interval at which a BGP neighbor session is reestablished by a device when the number of prefixes that have been received from a BGP peer has exceeded the maximum prefix limit.

The network operator can configure a device running BGP to automatically reestablish a neighbor session that has been brought down because the configured maximum-prefix limit has been exceeded. No intervention from the network operator is required when this feature is enabled.

Note This task attempts to reestablish a disabled BGP neighbor session at the configured time interval that is specified by the network operator. However, the configuration of the restart timer alone cannot change or correct a peer that is sending an excessive number of prefixes. The network operator will need to reconfigure the maximum-prefix limit or reduce the number of prefixes that are sent from the peer. A peer that is configured to send too many prefixes can cause instability in the network, where an excessive number of prefixes are rapidly advertised and withdrawn. In this case, the **warning-only** keyword of the **neighbor maximum-prefix** command can be configured to disable the restart capability while the network operator corrects the underlying problem.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **peer-group**
5. **neighbor** {*ip-address* | *ipv6-address***%** | *peer-group-name*} **peer-group** *peer-group-name*
6. **neighbor** {*ip-address* | *ipv6-address***%** | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]
7. **neighbor** {*ip-address* | *ipv6-address***%** | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]
8. **neighbor** {*ip-address* | *ipv6-address***%** | } **maximum-prefix** *maximum* [*threshold*] [**restart** *minutes*] [**warning-only**]
9. **end**
10. **show ip bgp neighbors** *ip-address*

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 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **peer-group** **Example:** `Device(config-router)# neighbor internal peer-group`	Creates a BGP or multiprotocol BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address***%** | *peer-group-name*} **peer-group** *peer-group-name* **Example:** `Device(config-router)# neighbor 10.4.9.5 peer-group internal`	Configures a BGP neighbor to member of a peer group. • **%** keyword is the IPv6 link-local address identifier. This keyword needs to be added whenever a link-local IPv6 address is used outside the context of its interface.
Step 6	**neighbor** {*ip-address* | *ipv6-address***%** | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `Device(config-router)# neighbor internal remote-as 100`	Adds a peer group to the BGP or multiprotocol BGP neighbor table.
Step 7	**neighbor** {*ip-address* | *ipv6-address***%** | *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]	Adds an entry to the BGP or multiprotocol BGP neighbor table.

	Command or Action	Purpose
	Example: `Device(config-router)# neighbor 10.4.9.5 remote-as 100`	
Step 8	**neighbor** {*ip-address* | *ipv6-address***%** | } **maximum-prefix** *maximum* [*threshold*] [**restart** *minutes*] [**warning-only**] **Example:** `Device(config-router)# neighbor 10.4.9.5 maximum-prefix 1000 90 restart 60`	Configures the maximum-prefix limit on a router that is running BGP. • Use the **restart** keyword and *minutes* argument to configure the router to automatically reestablish a neighbor session that has been disabled because the maximum-prefix limit has been exceeded. The configurable range of *minutes* is from 1 to 65535 minutes. • Use the **warning-only** keyword to configure the device to disable the restart capability to allow you to adjust a peer that is sending too many prefixes. **Note** If the *minutes* argument is not configured, the disabled session will stay down after the maximum-prefix limit is exceeded. This is the default behavior.
Step 9	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 10	**show ip bgp neighbors** *ip-address* **Example:** `Device# show ip bgp neighbors 10.4.9.5`	(Optional) Displays information about the TCP and BGP connections to neighbors. • In this example, the output from this command will display the maximum prefix limit for the specified neighbor and the configured restart timer value.

Examples

The following sample output from the **show ip bgp neighbors** command verifies that a device has been configured to automatically reestablish disabled neighbor sessions. The output shows that the maximum prefix limit for neighbor 10.4.9.5 is set to 1000 prefixes, the restart threshold is set to 90 percent, and the restart interval is set at 60 minutes.

```
Device# show ip bgp neighbors 10.4.9.5

BGP neighbor is 10.4.9.5,  remote AS 101, internal link
  BGP version 4, remote router ID 10.4.9.5
  BGP state = Established, up for 2w2d
  Last read 00:00:14, hold time is 180, keepalive interval is 60 seconds
  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:         23095      23095
    Route Refresh:          0          0
    Total:              23096      23096
  Default minimum time between advertisement runs is 5 seconds
 For address family: IPv4 Unicast
  BGP table version 1, neighbor versions 1/0 1/0
  Output queue sizes : 0 self, 0 replicated
  Index 2, Offset 0, Mask 0x4
  Member of update-group 2
                                Sent       Rcvd
  Prefix activity:              ----       ----
    Prefixes Current:              0          0
    Prefixes Total:                0          0
    Implicit Withdraw:             0          0
    Explicit Withdraw:             0          0
    Used as bestpath:            n/a          0
    Used as multipath:           n/a          0
                                  Outbound    Inbound
  Local Policy Denied Prefixes:   --------    -------
    Total:                                0          0
!Configured maximum number of prefixes and restart interval information!
  Maximum prefixes allowed 1000
  Threshold for warning message 90%, restart interval 60 min
  Number of NLRIs in the update sent: max 0, min 0
  Connections established 1; dropped 0
  Last reset never
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Local host: 10.4.9.21, Local port: 179
Foreign host: 10.4.9.5, Foreign port: 11871
Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)
Event Timers (current time is 0x5296BD2C):
Timer          Starts    Wakeups            Next
Retrans         23098          0             0x0
TimeWait            0          0             0x0
AckHold         23096      22692             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0
iss: 1900546793  snduna: 1900985663  sndnxt: 1900985663     sndwnd:  14959
irs: 2894590641  rcvnxt: 2895029492  rcvwnd:      14978  delrcvwnd:   1406
SRTT: 300 ms, RTTO: 607 ms, RTV: 3 ms, KRTT: 0 ms
minRTT: 0 ms, maxRTT: 316 ms, ACK hold: 200 ms
Flags: passive open, nagle, gen tcbs
Datagrams (max data segment is 1460 bytes):
Rcvd: 46021 (out of order: 0), with data: 23096, total data bytes: 438850
Sent: 46095 (retransmit: 0, fastretransmit: 0), with data: 23097, total data by9
```

Troubleshooting Tips

Use the **clear ip bgp** command to reset a BGP connection using BGP soft reconfiguration. This command can be used to clear stored prefixes to prevent a device that is running BGP from exceeding the maximum-prefix limit.

Display of the following error messages can indicate an underlying problem that is causing the neighbor session to become disabled. You should check the values configured for the **neighbor maximum-prefix**

command and the configuration of any peers that are sending an excessive number of prefixes. The following sample error messages are similar to the error messages that may be displayed:

```
00:01:14:%BGP-5-ADJCHANGE:neighbor 10.10.10.2 Up
00:01:14:%BGP-4-MAXPFX:No. of unicast prefix received from 10.10.10.2 reaches 5, max 6
00:01:14:%BGP-3-MAXPFXEXCEED:No.of unicast prefix received from 10.10.10.2:7 exceed limit6
00:01:14:%BGP-5-ADJCHANGE:neighbor 10.10.10.2 Down - BGP Notification sent
00:01:14:%BGP-3-NOTIFICATION:sent to neighbor 10.10.10.2 3/1 (update malformed) 0 byte
```

The **bgp dampening** command can be used to configure the dampening of a flapping route or interface when a peer is sending too many prefixes and causing network instability. Use this command only when troubleshooting or tuning a device that is sending an excessive number of prefixes. For more details about BGP route dampening, see the "Configuring Advanced BGP Features" module.

Configuration Example for BGP Restart Neighbor Session After Max-Prefix Limit Reached

Example: Configuring a Router to Reestablish a Neighbor Session After the Maximum Prefix Limit Reached

The following example sets the maximum number of prefixes allowed from the neighbor at 192.168.6.6 to 2000 and configures the device to reestablish a peering session after 30 minutes if one has been disabled:

```
Device(config)# router bgp 101
Device(config-router)# neighbor internal peer-group
Device(config-router)# neighbor 10.4.9.5 peer-group internal
Device(config-router)# neighbor internal remote-as 100
Device(config-router)# neighbor 10.4.9.5 remote-as 100
Device(config-router)# neighbor 10.4.9.5 maximum-prefix 2000 90 restart 30
Device(config-router)# end
```

Additional References for BGP Restart Neighbor Session After Max-Prefix Limit Reached

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*
RFC 4486	*Subcodes for BGP Cease Notification Message*

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

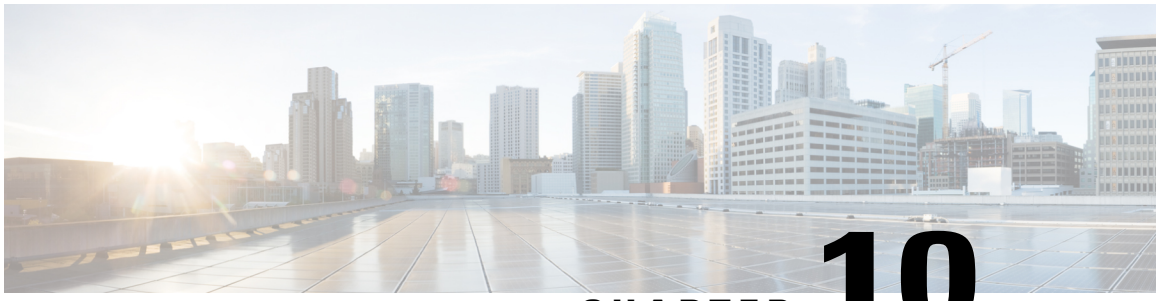
Feature Information for BGP Restart Neighbor Session after Max-Prefix Limit

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 Restart Session After Max-Prefix Limit

Feature Name	Releases	Feature Information
BGP Restart Session After Max-Prefix Limit	Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE	The BGP Restart Session After Max-Prefix Limit Reached feature adds the **restart** keyword to the **neighbor maximum-prefix** command. This allows a network operator to configure the time interval at which a peering session is reestablished by a device when the number of prefixes that have been received from a peer has exceeded the maximum prefix limit. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches. The following commands were modified: **neighbor maximum-prefix** and **show ip bgp neighbors**.
BGP—Subcodes for BGP Cease Notification		Support for subcodes for BGP cease notification has been added.



CHAPTER 10

BGP 4 Soft Configuration

BGP4 soft configuration allows BGP4 policies to be configured and activated without clearing the BGP session, hence without invalidating the forwarding cache.

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 4 Soft Configuration

BGP Session Reset

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

- Hard reset—A hard reset tears down the specified peering sessions including the TCP connection and deletes routes coming from the specified peer.

- Soft reset—A soft reset uses stored prefix information to reconfigure and activate BGP routing tables without tearing down existing peering sessions. Soft reconfiguration uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply new BGP policy without disrupting the network. Soft reconfiguration can be configured for inbound or outbound sessions.
- Dynamic inbound soft reset—The route refresh capability, as defined in RFC 2918, allows the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers. The route refresh capability does not store update information locally for nondisruptive policy changes. It instead relies on dynamic exchange with supporting peers. Route refresh must first be advertised through BGP capability negotiation between peers. All BGP devices must support the route refresh capability. To determine if a BGP device supports this capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the device supports the route refresh capability:

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

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

How to Configure BGP 4 Soft Configuration

Configuring Inbound Soft Reconfiguration When Route Refresh Capability Is Missing

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp log-neighbor-changes**
5. **bgp soft-reconfig-backup**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]
8. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
9. Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.
10. **exit**
11. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address*
13. **end**
14. **show ip bgp neighbors** [*neighbor-address*]
15. **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 for the specified routing process.
Step 4	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.

	Command or Action	Purpose
Step 5	**bgp soft-reconfig-backup** **Example:** `Device(config-router)# bgp soft-reconfig-backup`	Configures a BGP speaker to perform inbound soft reconfiguration for peers that do not support the route refresh capability. • This command is used to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**] **Example:** `Device(config-router)# neighbor 192.168.1.2 soft-reconfiguration inbound`	Configures the Cisco software to start storing updates. • All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is done later, the stored information will be used to generate a new set of inbound updates.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router)# neighbor 192.168.1.2 route-map LOCAL in`	Applies a route map to incoming or outgoing routes. • In this example, the route map named LOCAL will be applied to incoming routes.
Step 9	Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.	—
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LOCAL permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named LOCAL is created.

	Command or Action	Purpose
Step 12	**set ip next-hop** *ip-address* **Example:** `Device(config-route-map)# set ip next-hop 192.168.1.144`	Specifies where output packets that pass a match clause of a route map for policy routing. • In this example, the ip address is set to 192.168.1.144.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 15	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

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

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

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

```
BGP routing table entry for 172.17.1.0/24, version 11
Paths: (3 available, best #3, table Default-IP-Routing-Table, RIB-failure(4))
Flag: 0x820
 Advertised to update-groups:
    1
 50000
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external
```

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

Configuration Examples for BGP 4 Soft Configuration

Examples: BGP Soft Reset

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

Example: Dynamic Inbound Soft Reset

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

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

Example: Inbound Soft Reset Using Stored Information

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

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

The following example clears the session with the neighbor 192.168.1.1:

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

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

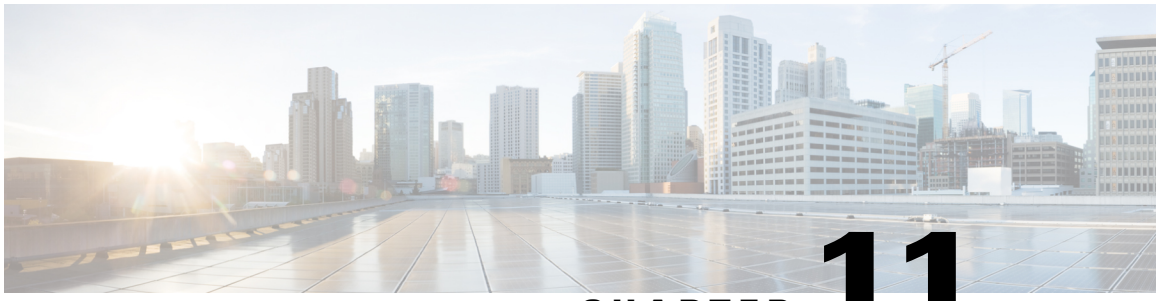
Feature Information for BGP 4 Soft Configuration

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 4 Soft Configuration

Feature Name	Releases	Feature Information
BGP 4 Soft Configuration	Cisco IOS XE Release 3.3SE	BGP 4 Soft Configuration allows BGP4 policies to be configured and activated without clearing the BGP session, hence without invalidating the forwarding cache. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 11

BGP Soft Reset

BGP Soft Reset feature provides automatic support for dynamic soft reset of inbound BGP routing table updates that is not dependent upon stored routing table update information. The new method requires no preconfiguration (as with the **neighbor soft-reconfiguration** command) and requires much less memory than the previous soft reset method for inbound routing table updates.

Finding Feature Information

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

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

Information About BGP Soft Reset

BGP Session Reset

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

- Hard reset—A hard reset tears down the specified peering sessions including the TCP connection and deletes routes coming from the specified peer.
- Soft reset—A soft reset uses stored prefix information to reconfigure and activate BGP routing tables without tearing down existing peering sessions. Soft reconfiguration uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply new BGP policy without disrupting the network. Soft reconfiguration can be configured for inbound or outbound sessions.
- Dynamic inbound soft reset—The route refresh capability, as defined in RFC 2918, allows the local device to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers. The route refresh capability does not store update information locally for nondisruptive policy changes. It instead relies on dynamic exchange with supporting peers. Route refresh must first be advertised through BGP capability negotiation between peers. All BGP devices must support the route refresh capability. To determine if a BGP device supports this capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the device supports the route refresh capability:

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

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

Routing Policy Change Management

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

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

Table 16: Advantages and Disadvantages of Hard and Soft Resets

Type of Reset	Advantages	Disadvantages
Hard reset	No memory overhead.	The prefixes in the BGP, IP, and Forwarding Information Base (FIB) tables provided by the neighbor are lost. A hard reset is not recommended.
Outbound soft reset	No configuration, and no storing of routing table updates.	Does not reset inbound routing table updates.

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

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

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

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

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

How to Configure BGP Soft Reset

Performing BGP Dynamic Inbound Soft Reset

SUMMARY STEPS

1. **enable**
2. **clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} **soft in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} **soft in** **Example:** `Device# clear ip bgp 192.168.1.2 soft in`	Performs a dynamic soft reset on the connection specified.

Performing BGP Outbound Soft Reset

SUMMARY STEPS

1. **enable**
2. **clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} **soft out**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip bgp** {***** | *autonomous-system-number* | *neighbor-address*} **soft out** **Example:** `Device# clear ip bgp 192.168.1.2 soft out`	Performs an outbound soft reset on the connection specified.

Configuring Inbound Soft Reconfiguration When Route Refresh Capability Is Missing

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp log-neighbor-changes**
5. **bgp soft-reconfig-backup**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**]
8. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
9. Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.
10. **exit**
11. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
12. **set ip next-hop** *ip-address*
13. **end**
14. **show ip bgp neighbors** [*neighbor-address*]
15. **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.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 5	**bgp soft-reconfig-backup** **Example:** `Device(config-router)# bgp soft-reconfig-backup`	Configures a BGP speaker to perform inbound soft reconfiguration for peers that do not support the route refresh capability. • This command is used to configure BGP to perform inbound soft reconfiguration for peers that do not support the route refresh capability. The configuration of this command allows you to configure BGP to store updates (soft reconfiguration) only as necessary. Peers that support the route refresh capability are unaffected by the configuration of this command.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **soft-reconfiguration** [**inbound**] **Example:** `Device(config-router)# neighbor 192.168.1.2 soft-reconfiguration inbound`	Configures the Cisco software to start storing updates. • All the updates received from this neighbor will be stored unmodified, regardless of the inbound policy. When inbound soft reconfiguration is done later, the stored information will be used to generate a new set of inbound updates.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router)# neighbor 192.168.1.2 route-map LOCAL in`	Applies a route map to incoming or outgoing routes. • In this example, the route map named LOCAL will be applied to incoming routes.
Step 9	Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.	—

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LOCAL permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named LOCAL is created.
Step 12	**set ip next-hop** *ip-address* **Example:** `Device(config-route-map)# set ip next-hop 192.168.1.144`	Specifies where output packets that pass a match clause of a route map for policy routing. • In this example, the ip address is set to 192.168.1.144.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 15	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

The following partial output from the **show ip bgp neighbors** command shows information about the TCP and BGP connections to the BGP neighbor 192.168.3.2. This peer does not support route refresh so the

soft-reconfig inbound paths for BGP peer 192.168.3.2 will be stored because there is no other way to update any inbound policy updates.

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

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

```
BGP routing table entry for 172.17.1.0/24, version 11
Paths: (3 available, best #3, table Default-IP-Routing-Table, RIB-failure(4))
Flag: 0x820
 Advertised to update-groups:
    1
 50000
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external
 50000, (received-only)
   192.168.3.2 from 192.168.3.2 (172.17.1.0)
     Origin incomplete, metric 0, localpref 100, valid, external
 40000
   192.168.1.2 from 192.168.1.2 (172.16.1.0)
     Origin incomplete, metric 0, localpref 200, valid, external, best
```

Configuration Examples for BGP Soft Reset

Examples: BGP Soft Reset

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

Example: Dynamic Inbound Soft Reset

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

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

Example: Inbound Soft Reset Using Stored Information

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

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

The following example clears the session with the neighbor 192.168.1.1:

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

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

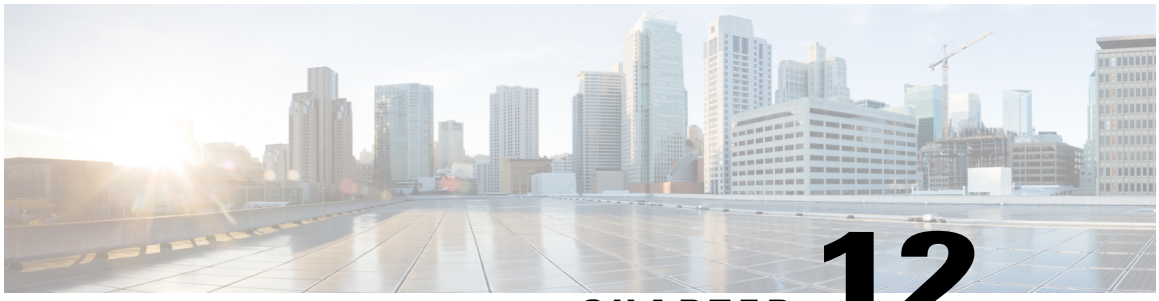
Feature Information for BGP Soft Reset

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 Soft Reset

Feature Name	Releases	Feature Information
BGP Soft Reset	Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE	BGP Soft Reset feature provides automatic support for dynamic soft reset of inbound BGP routing table updates that is not dependent upon stored routing table update information. The new method requires no preconfiguration (as with the **neighbor soft-reconfiguration** command) and requires much less memory than the previous soft reset method for inbound routing table updates. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 12

BGP Named Community Lists

The BGP Named Community Lists feature allows the network operator to assign meaningful names to community lists and increases the number of community lists that can be configured.

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 Named Community Lists

BGP COMMUNITIES Attribute

A BGP community is a group of routes that share a common property, regardless of their network, autonomous system, or any physical boundaries. In large networks, applying a common routing policy by using prefix lists or access lists requires individual peer statements on each networking device. Using the BGP COMMUNITIES attribute, BGP speakers with common routing policies can implement inbound or outbound route filters based on the community tag, rather than consult long lists of individual permit or deny statements. A COMMUNITIES attribute can contain multiple communities.

A route can belong to multiple communities. The network administrator defines the communities to which a route belongs. By default, all routes belong to the general Internet community.

In addition to numbered communities, there are several predefined (well-known) communities:

- no-export—Do not advertise this route to external BGP peers.
- no-advertise—Do not advertise this route to any peer.
- internet—Advertise this route to the Internet community. All BGP-speaking networking devices belong to this community.
- local-as—Do not send this route outside the local autonomous system.
- gshut—Community of routes gracefully shut down.

The COMMUNITIES attribute is optional, which means that it will not be passed on by networking devices that do not understand communities. Networking devices that understand communities must be configured to handle the communities or else the COMMUNITIES attribute will be discarded. By default, no COMMUNITIES attribute is sent to a neighbor. In order for a COMMUNITIES attribute to be sent to a neighbor, use the **neighbor send-community** command.

BGP Community Lists

A BGP community list is used to create groups of communities to use in a match clause of a route map. A community list can be used to control which routes are accepted, preferred, distributed, or advertised, for example. You can also use a community list to set, append, or modify the communities of a route.

- Standard community lists are used to specify well-known communities and community numbers.
- Expanded community lists are used to filter communities using a regular expression. Regular expressions are used to specify patterns to match COMMUNITIES attributes.

A BGP named community list allows you to assign a meaningful name to a community list. A named community list can be configured using community numbers, well-known communities, or regular expressions. All the rules of numbered community lists apply to named community lists, except that there is no limit on the number of named community lists that can be configured.

Note A maximum of 100 standard community lists and 100 expanded community lists can be configured. A named community list does not have this limitation.

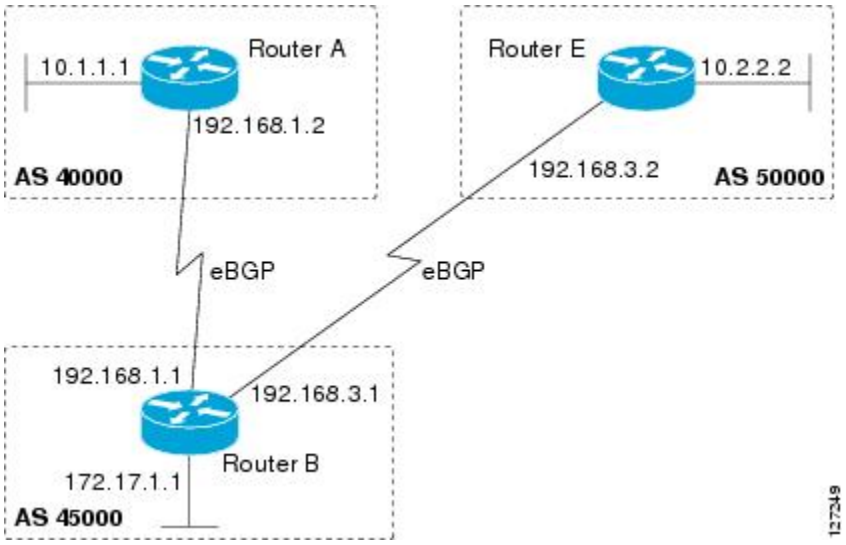
How to Use BGP Named Community Lists

Filtering Traffic Using Community Lists

Perform this task to filter traffic by creating a BGP community list, referencing the community list within a route map, and then applying the route map to a neighbor.

In this task, Router B in the figure below is configured with route maps and a community list to control incoming routes.

Figure 11: Topology for Which a Community List Is Configured



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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *route-map-name* {**in** | **out**}
7. **exit**
8. **exit**
9. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
10. **match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}
11. **set weight** *weight*
12. **exit**
13. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
14. **match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]}
15. **set community** *community-number*
16. **exit**
17. **ip community-list** {*standard-list-number* | **standard** *list-name* {**deny** | **permit**} [*community-number*] [*AA*:*NN*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]} | {*expanded-list-number* | **expanded** *list-name* {**deny** | **permit**} *regular-expression*}
18. Repeat Step 17 to create all the required community lists.
19. **exit**
20. **show ip community-list** [*standard-list-number* | *expanded-list-number* | *community-list-name*] [**exact-match**]

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 for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address or peer group name of the neighbor to the specified autonomous system BGP neighbor table of the local router.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the 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 6	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *route-map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.3.2 route-map 2000 in`	Applies a route map to inbound or outbound routes. • In this example, the route map called 2000 is applied to inbound routes from the BGP peer at 192.168.3.2.
Step 7	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 8	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 9	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*]	Creates a route map and enters route map configuration mode. • In this example, the route map called 2000 is defined.

	Command or Action	Purpose
	Example: `Device(config)# route-map 2000 permit 10`	
Step 10	**match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]} **Example:** `Device(config-route-map)# match community 1`	Matches on the communities in a BGP community list. • In this example, the route's community attribute is matched to communities in community list 1.
Step 11	**set weight** *weight* **Example:** `Device(config-route-map)# set weight 30`	Sets the weight of BGP routes that match the community list. • In this example, any route that matches community list 1 will have its weight set to 30.
Step 12	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 13	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map 3000 permit 10`	Creates a route map and enters route map configuration mode. • In this example, the route map called 3000 is defined.
Step 14	**match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]} **Example:** `Device(config-route-map)# match community 2`	Matches on the communities in a BGP community list. • In this example, the route's COMMUNITIES attribute is matched to communities in community list 2.
Step 15	**set community** *community-number* **Example:** `Device(config-route-map)# set community 99`	Sets the BGP communities attribute. • In this example, any route that matches community list 2 will have the COMMUNITIES attribute set to 99.
Step 16	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 17	**ip community-list** {*standard-list-number* | **standard** *list-name* {**deny** | **permit**} [*community-number*]	Creates a community list for BGP and controls access to it.

	Command or Action	Purpose

	[*AA*:*NN*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]} | {*expanded-list-number* | **expanded** *list-name* {**deny** | **permit**} *regular-expression*}
	
	Example:
	
	`Device(config)# ip community-list 1 permit 100`
	
	Example:
	
	`Device(config)# ip community-list 2 permit internet`
		• In the first example, community list 1 permits routes with a COMMUNITIES attribute of 100. Router E routes all have a COMMUNITIES attribute of 100, so their weight will be set to 30.
		• In the second example, community list 2 effectively permits all routes by specifying the **internet** community. Any routes that did not match community list 1 are checked against community list 2. All routes are permitted, but no changes are made to the route attributes.
		
		Note Two examples are shown here because the task example requires both of these statements to be configured.

Step 18	Repeat Step 17 to create all the required community lists.	—
Step 19	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and enters privileged EXEC mode.
Step 20	**show ip community-list** [*standard-list-number* | *expanded-list-number* | *community-list-name*] [**exact-match**] **Example:** `Device# show ip community-list 1`	Displays configured BGP community list entries.

Examples

The following sample output verifies that community list 1 has been created and it permits routes that have a community attribute of 100:

```
Device# show ip community-list 1

Community standard list 1
     permit 100
```

The following sample output verifies that community list 2 has been created and it effectively permits all routes by specifying the **internet** community:

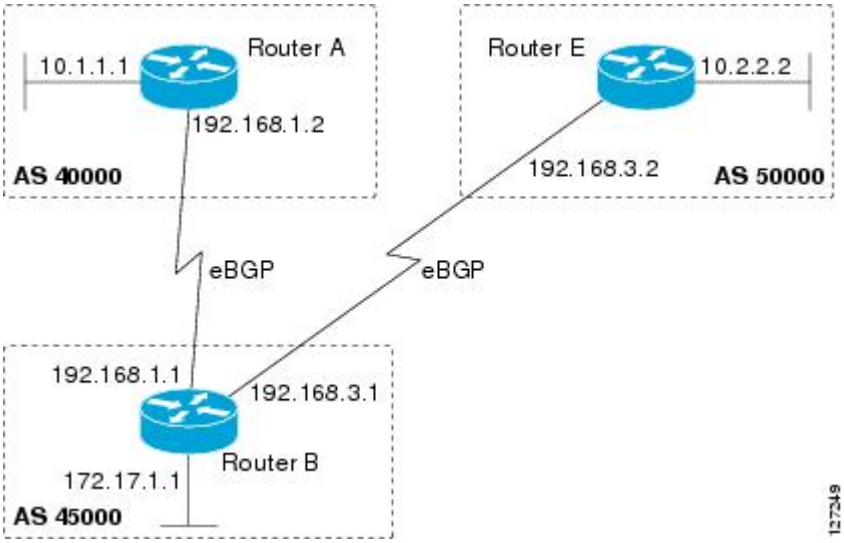
```
Device# show ip community-list 2

Community standard list 2
     permit internet
```

Filtering Traffic Using Extended Community Lists

Perform this task to filter traffic by creating an extended BGP community list to control outbound routes.

Figure 12: Topology for Which a Community List Is Configured



In this task, Router B in the figure above is configured with an extended named community list to specify that the BGP peer at 192.168.1.2 is not sent advertisements about any path through or from autonomous system 50000. The IP extended community-list configuration mode is used and the ability to resequence entries is shown.

Note A sequence number is applied to all extended community list entries by default, regardless of the configuration mode. Explicit sequencing and resequencing of extended community list entries can be configured only in IP extended community-list configuration mode, not in global configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip extcommunity-list** {*expanded-list-number* | **expanded** *list-name* | *standard-list-number* | **standard** *list-name*}
4. [*sequence-number*] {**deny** [*regular-expression*] | **exit** | **permit** [*regular-expression*]}
5. Repeat Step 4 for all the required permit or deny entries in the extended community list.
6. **resequence** [*starting-sequence*] [*sequence-increment*]
7. **exit**
8. **router bgp** *autonomous-system-number*
9. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
10. Repeat the prior step for all of the required BGP peers.
11. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
12. **network** *network-number* [**mask** *network-mask*]
13. **end**
14. **show ip extcommunity-list** [*list-name*]

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 extcommunity-list** {*expanded-list-number* | **expanded** *list-name* | *standard-list-number* | **standard** *list-name*} **Example:** `Device(config)# ip extcommunity-list expanded DENY50000`	Enters IP extended community-list configuration mode to create or configure an extended community list. • In this example, the expanded community list DENY50000 is created.
Step 4	[*sequence-number*] {**deny** [*regular-expression*] | **exit** | **permit** [*regular-expression*]}	Configures an expanded community list entry. • In the first example, an expanded community list entry with the sequence number 10 is configured to deny advertisements about paths from autonomous system 50000.

	Command or Action	Purpose

	Example:
	`Device(config-extcomm-list)# 10 deny _50000_`
	Example:
	`Device(config-extcomm-list)# 20 deny ^50000 .*`
		• In the second example, an expanded community list entry with the sequence number 20 is configured to deny advertisements about paths through autonomous system 50000.
		Note Two examples are shown here because the task example requires both these statements to be configured.
		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 5	Repeat Step 4 for all the required permit or deny entries in the extended community list.	—
Step 6	**resequence** [*starting-sequence*] [*sequence-increment*] **Example:** `Device(config-extcomm-list)# resequence 50 100`	Resequences expanded community list entries. • In this example, the sequence number of the first expanded community list entry is set to 50 and subsequent entries are set to increment by 100. The second expanded community list entry is therefore set to 150. **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 7	**exit** **Example:** `Device(config-extcomm-list)# exit`	Exits expanded community-list configuration mode and enters global configuration mode.
Step 8	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address or peer group name of the neighbor to the specified autonomous system BGP neighbor table of the local router.
Step 10	Repeat the prior step for all of the required BGP peers.	—
Step 11	**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 in the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. **Note** 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 12	**network** *network-number* [**mask** *network-mask*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols, the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates. **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 13	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.
Step 14	**show ip extcommunity-list** [*list-name*] **Example:** `Device# show ip extcommunity-list DENY50000`	Displays configured BGP expanded community list entries.

Examples

The following sample output verifies that the BGP expanded community list DENY50000 has been created, with the output showing that the entries to deny advertisements about autonomous system 50000 have been resequenced from 10 and 20 to 50 and 150:

```
Device# show ip extcommunity-list DENY50000

Expanded extended community-list DENY50000
     50 deny _50000_
     150 deny ^50000 .*
```

Configuration Examples for BGP Named Community Lists

Example: Filtering Traffic Using COMMUNITIES Attributes

This section contains two examples of the use of BGP COMMUNITIES attributes with route maps.

The first example configures a route map named *set-community*, which is applied to the outbound updates to the neighbor 172.16.232.50. The routes that pass access list 1 are given the well-known COMMUNITIES attribute value **no-export**. The remaining routes are advertised normally. The **no-export** community value automatically prevents the advertisement of those routes by the BGP speakers in autonomous system 200.

```
router bgp 100
 neighbor 172.16.232.50 remote-as 200
 neighbor 172.16.232.50 send-community
 neighbor 172.16.232.50 route-map set-community out
!
route-map set-community permit 10
 match address 1
 set community no-export
!
route-map set-community permit 20
 match address 2
```

The second example configures a route map named *set-community*, which is applied to the outbound updates to neighbor 172.16.232.90. All the routes that originate from autonomous system 70 have the COMMUNITIES attribute values 200 200 added to their already existing communities. All other routes are advertised as normal.

```
route-map bgp 200
 neighbor 172.16.232.90 remote-as 100
 neighbor 172.16.232.90 send-community
 neighbor 172.16.232.90 route-map set-community out
!
route-map set-community permit 10
 match as-path 1
 set community 200 200 additive
!
route-map set-community permit 20
!
ip as-path access-list 1 permit 70$
ip as-path access-list 2 permit .*
```

Additional References for BGP Named Community Lists

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 1997	*BGP Communities Attribute*

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

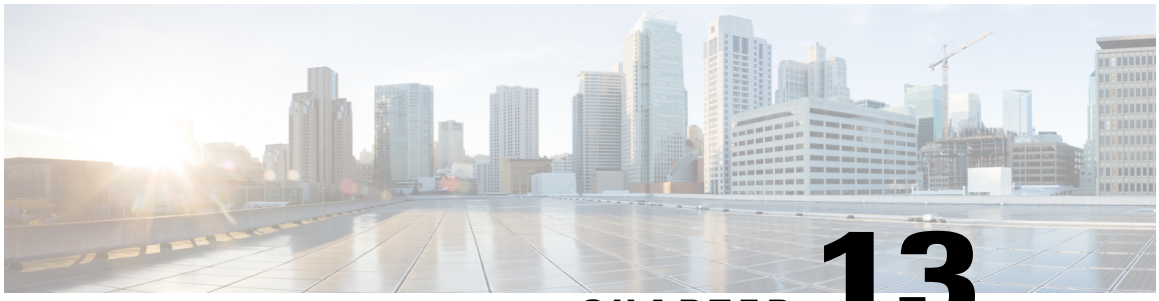
Feature Information for BGP Named Community Lists

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 Named Community Lists

Feature Name	Releases	Feature Information
BGP Named Community Lists	Cisco IOS XE Release 3.3SE	The BGP Named Community Lists feature introduces a new type of community list called the named community list. The BGP Named Community Lists feature allows the network operator to assign meaningful names to community lists and increases the number of community lists that can be configured. A named community list can be configured with regular expressions and with numbered community lists. All rules of numbered communities apply to named community lists except that there is no limitation on the number of community attributes that can be configured for a named community list. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 13

BGP 4 Prefix Filter and Inbound Route Maps

The BGP 4 Prefix Filter and Inbound Route Maps feature allows prefix-based matching support to the inbound neighbor route map. With this addition, an inbound route map can be used to enforce prefix-based policies.

Finding Feature Information

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

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

Information About BGP 4 Prefix Filter and Inbound Route Maps

BGP Policy Configuration

BGP policy configuration is used to control prefix processing by the BGP routing process and to filter routes from inbound and outbound advertisements. Prefix processing can be controlled by adjusting BGP timers, altering how BGP handles path attributes, limiting the number of prefixes that the routing process will accept, and configuring BGP prefix dampening. Prefixes in inbound and outbound advertisements are filtered using

route maps, filter lists, IP prefix lists, autonomous-system-path access lists, IP policy lists, and distribute lists. The table below shows the processing order of BGP policy filters.

Table 19: BGP Policy Processing Order

Inbound	Outbound
Route map	Distribute list
Filter list, AS-path access list, or IP policy	IP prefix list
IP prefix list	Filter list, AS-path access list, or IP policy
Distribute list	Route map

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

- Hard reset—A hard reset tears down the specified peering sessions, including the TCP connection, and deletes routes coming from the specified peer.
- Soft reset—A soft reset uses stored prefix information to reconfigure and activate BGP routing tables without tearing down existing peering sessions. Soft reset uses stored update information, at the cost of additional memory for storing the updates, to allow you to apply a new BGP policy without disrupting the network. Soft reset can be configured for inbound or outbound sessions.
- Dynamic inbound soft reset—The route refresh capability, as defined in RFC 2918, allows the local router to reset inbound routing tables dynamically by exchanging route refresh requests to supporting peers. The route refresh capability does not store update information locally for nondisruptive policy changes. It instead relies on dynamic exchange with supporting peers. Route refresh must first be advertised through BGP capability negotiation between peers. All BGP routers must support the route refresh capability.

To determine if a BGP router supports this capability, use the **show ip bgp neighbors** command. The following message is displayed in the output when the router supports the route refresh capability:

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

How to Configure BGP 4 Prefix Filter and Inbound Route Maps

Influencing Inbound Path Selection

BGP can be used to influence the choice of paths in another autonomous system. There may be several reasons for wanting BGP to choose a path that is not the obvious best route, for example, to avoid some types of transit traffic passing through an autonomous system or perhaps to avoid a very slow or congested link. BGP can influence inbound path selection using one of the following BGP attributes:

- AS-path

- Multi-Exit Discriminator (MED)

Perform one of the following tasks to influence inbound path selection:

Influencing Inbound Path Selection by Modifying the AS_PATH Attribute

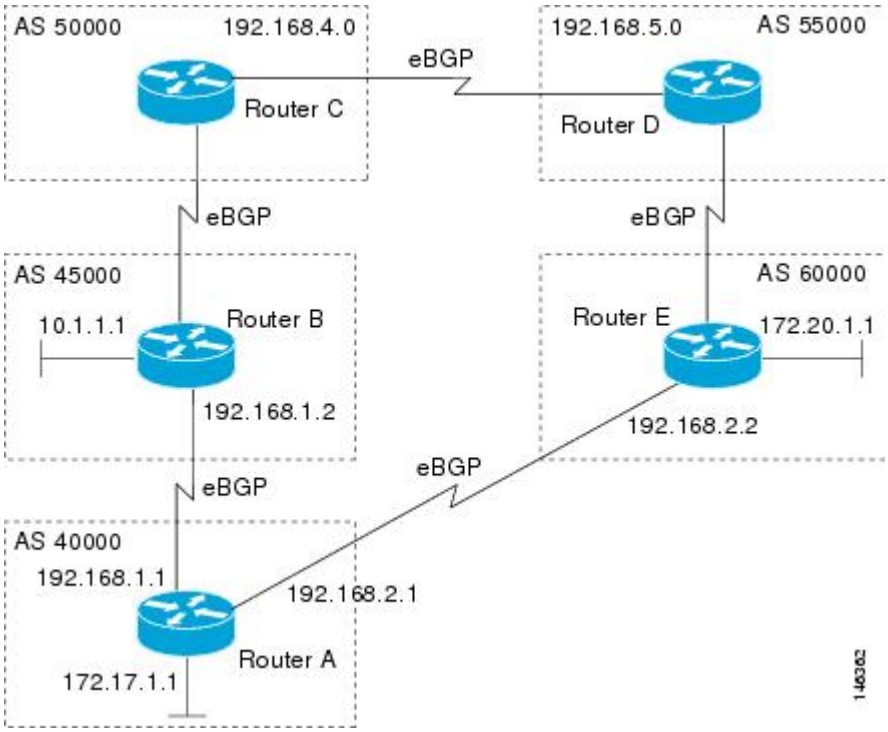
Perform this task to influence the inbound path selection for traffic destined for the 172.17.1.0 network by modifying the AS_PATH attribute. The configuration is performed at Router A in the figure below. For a configuration example of this task using 4-byte autonomous system numbers in asplain format, see the "Example: Influencing Inbound Path Selection by Modifying the AS_PATH Attribute Using 4-Byte AS Numbers".

One of the methods that BGP can use to influence the choice of paths in another autonomous system is to modify the AS_PATH attribute. For example, in the figure below, Router A advertises its own network, 172.17.1.0, to its BGP peers in autonomous system 45000 and autonomous system 60000. When the routing information is propagated to autonomous system 50000, the routers in autonomous system 50000 have network reachability information about network 172.17.1.0 from two different routes. The first route is from autonomous system 45000 with an AS_PATH consisting of 45000, 40000, the second route is through autonomous system 55000 with an AS-path of 55000, 60000, 40000. If all other BGP attribute values are the same, Router C in autonomous system 50000 would choose the route through autonomous system 45000 for traffic destined for network 172.17.1.0 because it is the shortest route in terms of autonomous systems traversed.

Autonomous system 40000 now receives all traffic from autonomous system 50000 for the 172.17.1.0 network through autonomous system 45000. If, however, the link between autonomous system 45000 and autonomous system 40000 is a really slow and congested link, the **set as-path prepend** command can be used at Router A to influence inbound path selection for the 172.17.1.0 network by making the route through autonomous system 45000 appear to be longer than the path through autonomous system 60000. The configuration is done at Router A in the figure below by applying a route map to the outbound BGP updates to Router B. Using the **set as-path prepend** command, all the outbound BGP updates from Router A to Router B will have their AS_PATH attribute modified to add the local autonomous system number 40000 twice. After the configuration, autonomous system 50000 receives updates about the 172.17.1.0 network through autonomous system 45000. The new AS_PATH is 45000, 40000, 40000, and 40000, which is now longer than the AS-path from autonomous system 55000 (unchanged at a value of 55000, 60000, 40000). Networking devices in autonomous

system 50000 will now prefer the route through autonomous system 55000 to forward packets with a destination address in the 172.17.1.0 network.

Figure 13: Network Topology for Modifying the AS_PATH Attribute



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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
7. **neighbor** {*ip-address* | *peer-group-name*} **activate**
8. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
9. **exit-address-family**
10. **exit**
11. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
12. **set as-path** {**tag** | **prepend** *as-path-string*}
13. **end**
14. **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:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 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 BGP peer on Router B at 192.168.1.2 is added to the IPv4 multiprotocol BGP neighbor table and will receive BGP updates.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The unicast keyword specifies the IPv4 unicast address family. By default, the 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 6	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	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.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.2 activate`	Enables address exchange for address family IPv4 unicast for the BGP neighbor at 192.168.1.2 on Router B.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.1.2 route-map PREPEND out`	Applies a route map to incoming or outgoing routes. • In this example, the route map named PREPEND is applied to outbound routes to Router B.
Step 9	**exit-address-family** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map PREPEND permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named PREPEND is created with a permit clause.
Step 12	**set as-path** {**tag** | **prepend** *as-path-string*} **Example:** `Device(config-route-map)# set as-path prepend 40000 40000`	Modifies an autonomous system path for BGP routes. • Use the **prepend** keyword to prepend an arbitrary autonomous system path string to BGP routes. Usually the local autonomous system number is prepended multiple times, increasing the autonomous system path length. • In this example, two additional autonomous system entries are added to the autonomous system path for outbound routes to Router B.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 14	**show running-config** **Example:** `Device# show running-config`	Displays the running configuration file.

Examples

The following partial output of the **show running-config** command shows the configuration from this task.

Router A

```
Device# show running-config
.
.
.
router bgp 40000
 neighbor 192.168.1.2 remote-as 45000
 !
 address-family ipv4
 neighbor 192.168.1.2 activate
 neighbor 192.168.1.2 route-map PREPEND out
 no auto-summary
 no synchronization
 network 172.17.1.0 mask 255.255.255.0
 exit-address-family
!
route-map PREPEND permit 10
 set as-path prepend 40000 40000
.
.
.
```

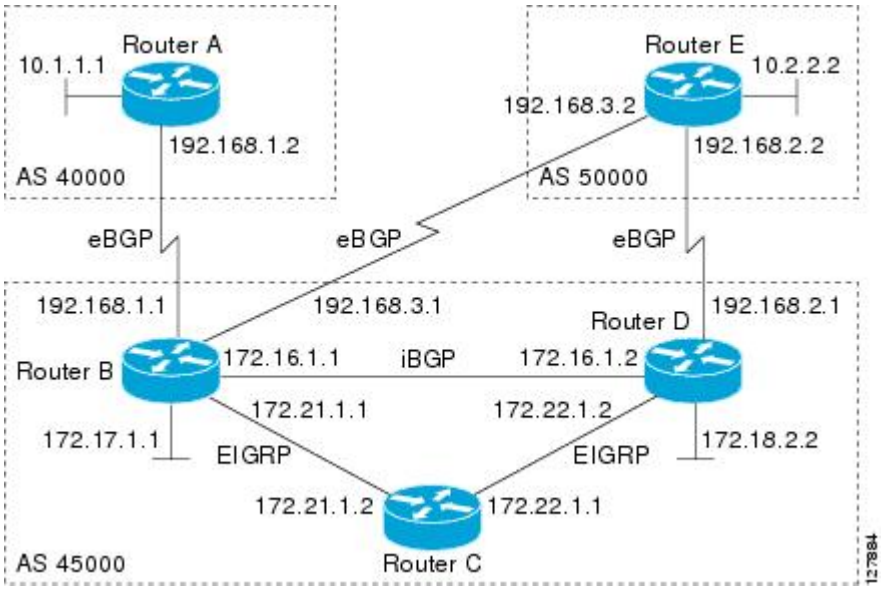
Influencing Inbound Path Selection by Setting the MED Attribute

One of the methods that BGP can use to influence the choice of paths into another autonomous system is to set the Multi-Exit Discriminator (MED) attribute. The MED attribute indicates (to an external peer) a preferred path to an autonomous system. If there are multiple entry points to an autonomous system, the MED can be used to influence another autonomous system to choose one particular entry point. A metric is assigned using route maps where a lower MED metric is preferred by the software over a higher MED metric.

Perform this task to influence inbound path selection by setting the MED metric attribute. The configuration is performed at Router B and Router D in the figure below. Router B advertises the network 172.16.1.0. to its BGP peer, Router E in autonomous system 50000. Using a simple route map Router B sets the MED metric to 50 for outbound updates. The task is repeated at Router D but the MED metric is set to 120. When Router E receives the updates from both Router B and Router D the MED metric is stored in the BGP routing table. Before forwarding packets to network 172.16.1.0, Router E compares the attributes from peers in the same

autonomous system (both Router B and Router D are in autonomous system 45000). The MED metric for Router B is less than the MED for Router D, so Router E will forward the packets through Router B.

Figure 14: Network Topology for Setting the MED Attribute



Use the **bgp always-compare-med** command to compare MED attributes from peers in other autonomous systems.

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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]
7. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
8. **exit**
9. **exit**
10. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
11. **set metric** *value*
12. **end**
13. Repeat Step 1 through Step 12 at Router D.
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 for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.3.2 remote-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.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the 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 6	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.16.1.0 mask 255.255.255.0`	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.

	Command or Action	Purpose
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.3.2 route-map MED out`	Applies a route map to incoming or outgoing routes. • In this example, the route map named MED is applied to outbound routes to the BGP peer at Router E.
Step 8	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 9	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 10	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map MED permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named MED is created.
Step 11	**set metric** *value* **Example:** `Device(config-route-map)# set metric 50`	Sets the MED metric value.
Step 12	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 13	Repeat Step 1 through Step 12 at Router D.	—
Step 14	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp 172.17.1.0 255.255.255.0`	(Optional) Displays the entries in the BGP routing table. • Use this command at Router E in the figure above when both Router B and Router D have configured the MED attribute. • 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 is from Router E in the figure above after this task has been performed at both Router B and Router D. Note the metric (MED) values for the two routes to network 172.16.1.0. The peer 192.168.2.1 at Router D has a metric of 120 for the path to network 172.16.1.0, whereas the peer 192.168.3.1 at Router B has a metric of 50. The entry for the peer 192.168.3.1 at Router B has the word best at the end of the entry to show that Router E will choose to send packets destined for network 172.16.1.0 via Router B because the MED metric is lower.

```
Device# show ip bgp 172.16.1.0

BGP routing table entry for 172.16.1.0/24, version 10
Paths: (2 available, best #2, table Default-IP-Routing-Table)
  Advertised to update-groups:
     1
  45000
    192.168.2.1 from 192.168.2.1 (192.168.2.1)
      Origin IGP, metric 120, localpref 100, valid, external
  45000
    192.168.3.1 from 192.168.3.1 (172.17.1.99)
      Origin IGP, metric 50, localpref 100, valid, external, best
```

Configuration Examples for BGP4 Prefix Filter and Inbound Route Maps

Example: Influencing Inbound Path Selection

The following example shows how you can use route maps to modify incoming data from a neighbor. Any route received from 10.222.1.1 that matches the filter parameters set in autonomous system access list 200 will have its weight set to 200 and its local preference set to 250, and it will be accepted.

```
router bgp 100
!
 neighbor 10.222.1.1 route-map FIX-WEIGHT in
 neighbor 10.222.1.1 remote-as 1
!
ip as-path access-list 200 permit ^690$
ip as-path access-list 200 permit ^1800
!
route-map FIX-WEIGHT permit 10
 match as-path 200
 set local-preference 250
 set weight 200
```

In the following example, the route map named FINANCE marks all paths originating from autonomous system 690 with an MED metric attribute of 127. The second permit clause is required so that routes not matching autonomous system path list 1 will still be sent to neighbor 10.1.1.1.

```
router bgp 65000
 neighbor 10.1.1.1 route-map FINANCE out
!
ip as-path access-list 1 permit ^690_
ip as-path access-list 2 permit .*
!
route-map FINANCE permit 10
 match as-path 1
 set metric 127
!
```

```
route-map FINANCE permit 20
 match as-path 2
```

Inbound route maps could perform prefix-based matching and set various parameters of the update. Inbound prefix matching is available in addition to autonomous system path and community list matching. The following example shows how the route map named SET-LOCAL-PREF sets the local preference of the inbound prefix 172.20.0.0/16 to 120:

```
!
router bgp 65100
 network 10.108.0.0
 neighbor 10.108.1.1 remote-as 65200
 neighbor 10.108.1.1 route-map SET-LOCAL-PREF in
!
route-map SET-LOCAL-PREF permit 10
 match ip address 2
 set local-preference 120
!
route-map SET-LOCAL-PREF permit 20
!
access-list 2 permit 172.20.0.0 0.0.255.255
access-list 2 deny any
```

Example: Influencing Inbound Path Selection by Modifying the AS-path Attribute Using 4-Byte AS Numbers

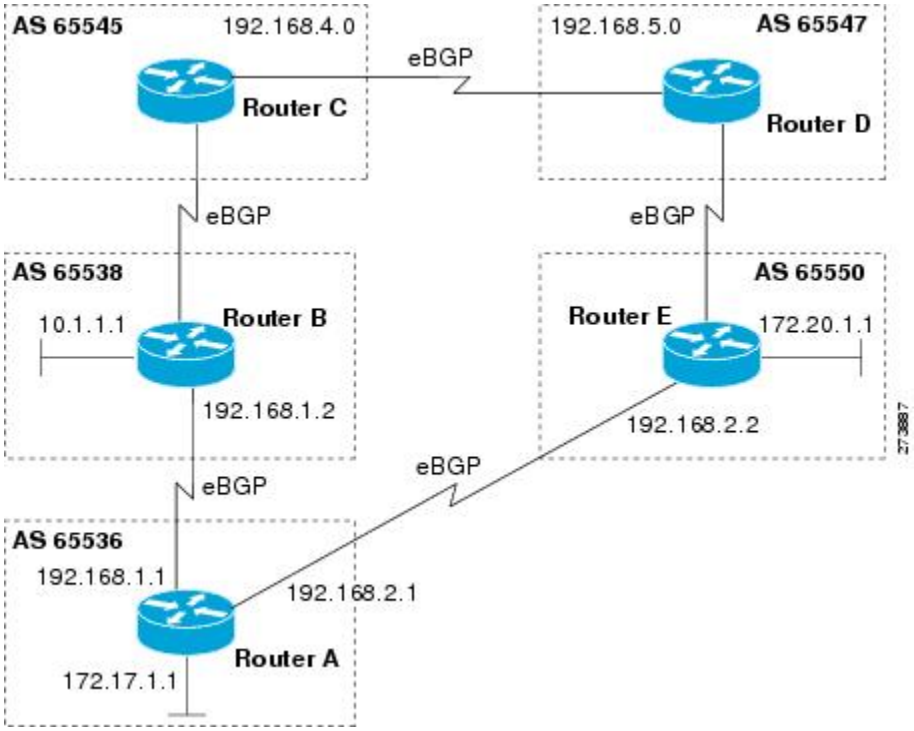
This example shows how to configure BGP to influence the inbound path selection for traffic destined for the 172.17.1.0 network by modifying the AS-path attribute. In Cisco IOS XE Release 2.4 and later releases, BGP support for 4-octet (4-byte) autonomous system numbers was introduced. The 4-byte autonomous system numbers in this example are formatted in the default asplain (decimal value) format; for example, Router B is in autonomous system number 65538 in the figure below.

One of the methods that BGP can use to influence the choice of paths in another autonomous system is to modify the AS-path attribute. For example, in the figure below, Router A advertises its own network, 172.17.1.0, to its BGP peers in autonomous system 65538 and autonomous system 65550. When the routing information is propagated to autonomous system 65545, the routers in autonomous system 65545 have network reachability information about network 172.17.1.0 from two different routes. The first route is from autonomous system 65538 with an AS-path consisting of 65538, 65536. The second route is through autonomous system 65547 with an AS-path of 65547, 65550, 65536. If all other BGP attribute values are the same, Router C in autonomous system 65545 would choose the route through autonomous system 65538 for traffic destined for network 172.17.1.0 because it is the shortest route in terms of autonomous systems traversed.

Autonomous system 65536 now receives all traffic from autonomous system 65545 for the 172.17.1.0 network through Router B in autonomous system 65538. If, however, the link between autonomous system 65538 and autonomous system 65536 is a really slow and congested link, the **set as-path prepend** command can be used at Router A to influence inbound path selection for the 172.17.1.0 network by making the route through autonomous system 65538 appear to be longer than the path through autonomous system 65550. The configuration is done at Router A in the figure below by applying a route map to the outbound BGP updates to Router B. Using the **set as-path prepend** command, all the outbound BGP updates from Router A to Router B will have their AS-path attribute modified to add the local autonomous system number 65536 twice. After the configuration, autonomous system 65545 receives updates about the 172.17.1.0 network through autonomous system 65538. The new AS-path is 65538, 65536, 65536, 65536, which is now longer than the AS-path from autonomous system 65547 (unchanged at a value of 65547, 65550, 65536). Networking devices in autonomous

system 65545 will now prefer the route through autonomous system 65547 to forward packets with a destination address in the 172.17.1.0 network.

Figure 15: Network Topology for Modifying the AS-path Attribute



The configuration for this example is performed at Router A in the figure above.

```
router bgp 65536
 address-family ipv4 unicast
  network 172.17.1.0 mask 255.255.255.0
  neighbor 192.168.1.2 remote-as 65538
  neighbor 192.168.1.2 activate
  neighbor 192.168.1.2 route-map PREPEND out
  exit-address-family
 exit
route-map PREPEND permit 10
 set as-path prepend 65536 65536
```

Example: Filtering BGP Prefixes Using a Single Prefix List

The following example shows how a prefix list denies the default route 0.0.0.0/0:

```
ip prefix-list abc deny 0.0.0.0/0
```

The following example shows how a prefix list permits a route that matches the prefix 10.0.0.0/8:

```
ip prefix-list abc permit 10.0.0.0/8
```

The following example shows how to configure the BGP process so that it accepts only prefixes with a prefix length of /8 to /24:

```
router bgp 40000
 network 10.20.20.0
 distribute-list prefix max24 in
!
ip prefix-list max24 seq 5 permit 0.0.0.0/0 ge 8 le 24
```

The following example configuration shows how to conditionally originate a default route (0.0.0.0/0) in RIP when a prefix 10.1.1.0/24 exists in the routing table:

```
ip prefix-list cond permit 10.1.1.0/24
!
route-map default-condition permit 10
 match ip address prefix-list cond
!
router rip
 default-information originate route-map default-condition
```

The following example shows how to configure BGP to accept routing updates from 192.168.1.1 only, besides filtering on the prefix length:

```
router bgp 40000
 distribute-list prefix max24 gateway allowlist in
!
ip prefix-list allowlist seq 5 permit 192.168.1.1/32
!
```

The following example shows how to direct the BGP process to filter incoming updates to the prefix using name1, and match the gateway (next hop) of the prefix being updated to the prefix list name2, on Gigabit Ethernet interface 0/0/0:

```
router bgp 103
 distribute-list prefix name1 gateway name2 in gigabitethernet 0/0/0
```

Additional References for BGP Restart Neighbor Session After Max-Prefix Limit Reached

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*

Standard/RFC	Title
RFC 4486	*Subcodes for BGP Cease Notification Message*

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

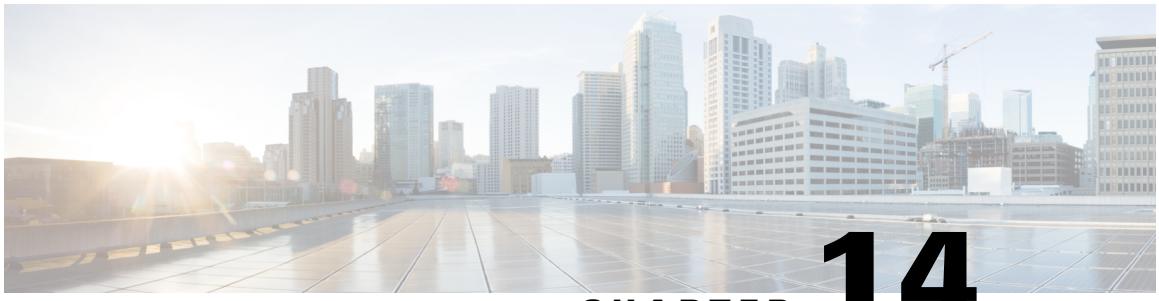
Feature Information for BGP 4 Prefix Filter and Inbound Route Maps

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 20: Feature Information for BGP 4 Prefix Filter and Inbound Route Maps

Feature Name	Releases	Feature Information
BGP 4 Prefix Filter and Inbound Route Maps	Cisco IOS XE Release 3.1.0SG Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE	The BGP 4 Prefix Filter and Inbound Route Maps feature allows prefix-based matching support to the inbound neighbor route map. With this addition, an inbound route map can be used to enforce prefix-based policies. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.



CHAPTER 14

BGP Prefix-Based Outbound Route Filtering

The BGP Prefix-Based Outbound Route Filtering (ORF) feature uses BGP ORF send and receive capabilities to minimize the number of BGP updates that are sent between BGP peers. Configuring this feature can help reduce the amount of system resources required for generating and processing routing updates by filtering out unwanted routing updates at the source. For example, this feature can be used to reduce the amount of processing required on a router that is not accepting full routes from a service provider network.

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 Prefix-Based Outbound Route Filtering

BGP Prefix-Based Outbound Route Filtering

BGP prefix-based outbound route filtering uses the BGP ORF send and receive capabilities to minimize the number of BGP updates that are sent between BGP peers. Configuring BGP ORF can help reduce the amount of system resources required for generating and processing routing updates by filtering out unwanted routing

updates at the source. For example, BGP ORF can be used to reduce the amount of processing required on a router that is not accepting full routes from a service provider network.

The BGP prefix-based outbound route filtering is enabled through the advertisement of ORF capabilities to peer routers. The advertisement of the ORF capability indicates that a BGP peer will accept a prefix list from a neighbor and apply the prefix list to locally configured ORFs (if any exist). When this capability is enabled, the BGP speaker can install the inbound prefix list filter to the remote peer as an outbound filter, which reduces unwanted routing updates.

The BGP prefix-based outbound route filtering can be configured with send or receive ORF capabilities. The local peer advertises the ORF capability in send mode. The remote peer receives the ORF capability in receive mode and applies the filter as an outbound policy. The local and remote peers exchange updates to maintain the ORF on each router. Updates are exchanged between peer routers by address family depending on the ORF prefix list capability that is advertised. The remote peer starts sending updates to the local peer after a route refresh has been requested with the **clear ip bgp in prefix-filter** command or after an ORF prefix list with immediate status is processed. The BGP peer will continue to apply the inbound prefix list to received updates after the local peer pushes the inbound prefix list to the remote peer.

How to Configure BGP Prefix-Based Outbound Route Filtering

Filtering Outbound Routes Based on BGP Prefix

Before You Begin

BGP peering sessions must be established, and BGP ORF capabilities must be enabled on each participating router before prefix-based ORF announcements can be received.

Note

- BGP prefix-based outbound route filtering does not support multicast.
- IP addresses that are used for outbound route filtering must be defined in an IP prefix list. BGP distribute lists and IP access lists are not supported.
- Outbound route filtering is configured on only a per-address family basis and cannot be configured under the general session or BGP routing process.
- Outbound route filtering is configured for external peering sessions only.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network / length* | **permit** *network / length*} [**ge** *ge-value*] [**le** *le-value*]
4. **router bgp** *autonomous-system-number*
5. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** *ip-address* **ebgp-multihop** [*hop-count*]
8. **neighbor** *ip-address* **capability orf prefix-list** [**send** | **receive** | **both**]
9. **neighbor** {*ip-address* | *peer-group-name*} **prefix-list** *prefix-list-name* {**in** | **out**}
10. **end**
11. **clear ip bgp** {*ip-address* | *****} **in prefix-filter**

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 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 FILTER seq 10 permit 192.168.1.0/24`	Creates a prefix list for prefix-based outbound route filtering. • Outbound route filtering supports prefix length matching, wildcard-based prefix matching, and exact address prefix matching on a per address-family basis. • The prefix list is created to define the outbound route filter. The filter must be created when the outbound route filtering capability is configured to be advertised in send mode or both mode. It is not required when a peer is configured to advertise receive mode only. • The example creates a prefix list named FILTER that defines the 192.168.1.0/24 subnet for outbound route filtering.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode, and creates a BGP routing process.

	Command or Action	Purpose
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the 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. **Note** Outbound route filtering is configured on a per-address family basis.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 10.1.1.1 remote-as 200`	Establishes peering with the specified neighbor or peer group. BGP peering must be established before ORF capabilities can be exchanged. • The example establishes peering with the 10.1.1.1 neighbor.
Step 7	**neighbor** *ip-address* **ebgp-multihop** [*hop-count*] **Example:** `Device(config-router-af)# neighbor 10.1.1.1 ebgp-multihop`	Accepts or initiates BGP connections to external peers residing on networks that are not directly connected.
Step 8	**neighbor** *ip-address* **capability orf prefix-list** [**send** | **receive** | **both**] **Example:** `Device(config-router-af)# neighbor 10.1.1.1 capability orf prefix-list both`	Enables the ORF capability on the local router, and enables ORF capability advertisement to the BGP peer specified with the *ip-address* argument. • The **send** keyword configures a router to advertise ORF send capabilities. • The **receive** keyword configures a router to advertise ORF receive capabilities. • The **both** keyword configures a router to advertise send and receive capabilities. • The remote peer must be configured to either send or receive ORF capabilities before outbound route filtering is enabled. • The example configures the router to advertise send and receive capabilities to the 10.1.1.1 neighbor.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **prefix-list** *prefix-list-name* {**in** | **out**}	Applies an inbound prefix-list filter to prevent distribution of BGP neighbor information.

	Command or Action	Purpose
	Example: `Device(config-router-af)# neighbor 10.1.1.1 prefix-list FILTER in`	• In this example, the prefix list named FILTER is applied to incoming advertisements from the 10.1.1.1 neighbor, which prevents distribution of the 192.168.1.0/24 subnet.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode, and enters privileged EXEC mode.
Step 11	**clear ip bgp** {*ip-address* | *****} **in prefix-filter** **Example:** `Device# clear ip bgp 10.1.1.1 in prefix-filter`	Clears BGP outbound route filters and initiates an inbound soft reset. • A single neighbor or all neighbors can be specified. **Note** The inbound soft refresh must be initiated with the **clear ip bgp** command in order for this feature to function.

Configuration Examples for BGP Prefix-Based Outbound Route Filtering

Example: Influencing Outbound Path Selection

The following example creates an outbound route filter and configures Router A (10.1.1.1) to advertise the filter to Router-B (172.16.1.2). An IP prefix list named FILTER is created to specify the 192.168.1.0/24 subnet for outbound route filtering. The ORF send capability is configured on Router A so that Router A can advertise the outbound route filter to Router B.

Router A Configuration (Sender)

```
ip prefix-list FILTER seq 10 permit 192.168.1.0/24
!
router bgp 65100
 address-family ipv4 unicast
 neighbor 172.16.1.2 remote-as 65200
 neighbor 172.16.1.2 ebgp-multihop
 neighbor 172.16.1.2 capability orf prefix-list send
 neighbor 172.16.1.2 prefix-list FILTER in
 end
```

Router B Configuration (Receiver)

The following example configures Router B to advertise the ORF receive capability to Router A. Router B will install the outbound route filter, defined in the FILTER prefix list, after ORF capabilities have been

exchanged. An inbound soft reset is initiated on Router B at the end of this configuration to activate the outbound route filter.

```
router bgp 65200
 address-family ipv4 unicast
 neighbor 10.1.1.1 remote-as 65100
 neighbor 10.1.1.1 ebgp-multihop 255
 neighbor 10.1.1.1 capability orf prefix-list receive
 end
clear ip bgp 10.1.1.1 in prefix-filter
```

The following example shows how the route map named set-as-path is applied to outbound updates to the neighbor 10.69.232.70. The route map will prepend the autonomous system path “65100 65100” to routes that pass access list 1. The second part of the route map is to permit the advertisement of other routes.

```
router bgp 65100
 network 172.16.0.0
 network 172.17.0.0
 neighbor 10.69.232.70 remote-as 65200
 neighbor 10.69.232.70 route-map set-as-path out
!
route-map set-as-path 10 permit
 match address 1
 set as-path prepend 65100 65100
!
route-map set-as-path 20 permit
 match address 2
!
access-list 1 permit 172.16.0.0 0.0.255.255
access-list 1 permit 172.17.0.0 0.0.255.255
!
access-list 2 permit 0.0.0.0 255.255.255.255
```

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 Prefix-Based Outbound Route Filtering

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 21: Feature Information for BGP Prefix-Based Outbound Route Filtering

Feature Name	Releases	Feature Information
BGP Prefix-Based Outbound Route Filtering	Cisco IOS XE Release 3.2SE Cisco IOS XE Release 3.3SE	The BGP Prefix-Based Outbound Route Filtering feature uses BGP ORF send and receive capabilities to minimize the number of BGP updates that are sent between BGP peers. Configuring this feature can help reduce the amount of system resources required for generating and processing routing updates by filtering out unwanted routing updates at the source. For example, this feature can be used to reduce the amount of processing required on a router that is not accepting full routes from a service provider network. In Cisco IOS XE Release 3.3SE, support was added for the Cisco Catalyst 3650 Series Switches and Cisco Catalyst 3850 Series Switches.