IP Routing: BGP Configuration Guide, Cisco IOS Release 15M&T

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 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.

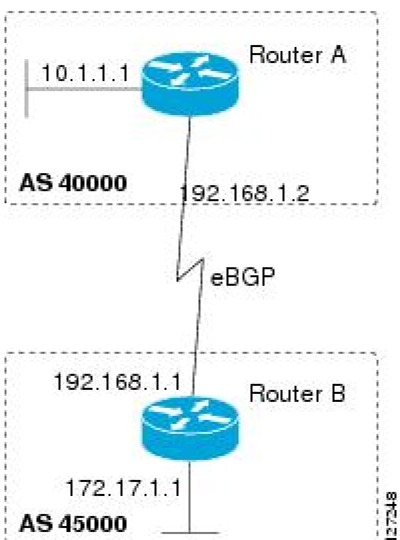
BGP Autonomous Systems

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.

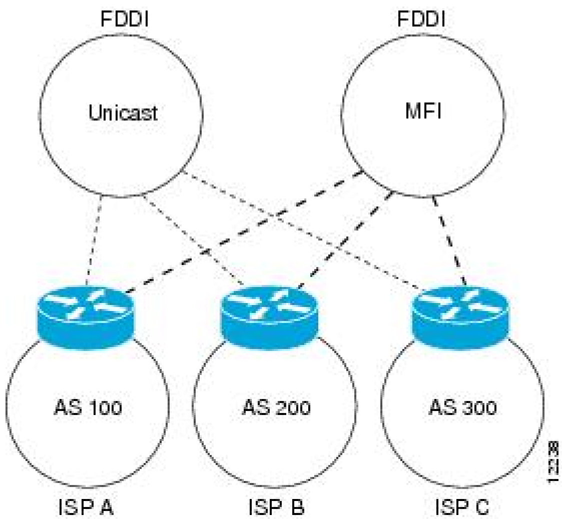
In BGP, the only way to perform interdomain multicast routing is to use the BGP infrastructure that is in place for unicast routing. If the routers are not multicast-capable, or if there are differing policies about where multicast traffic should flow, multicast routing cannot be supported without multiprotocol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

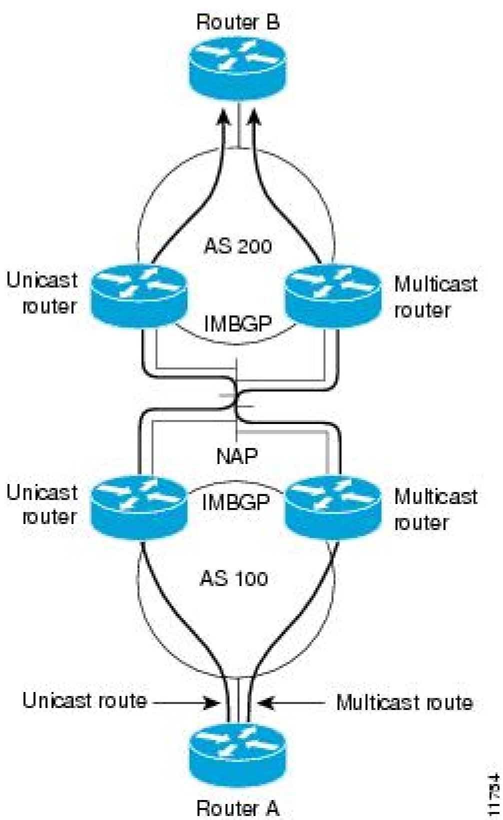


The figure below is a topology of unicast-only routers and multicast-only routers. The two routers on the left are unicast-only routers (that is, they do not support or are not configured to perform multicast routing). The two routers on the right are multicast-only routers. Routers A and B support both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

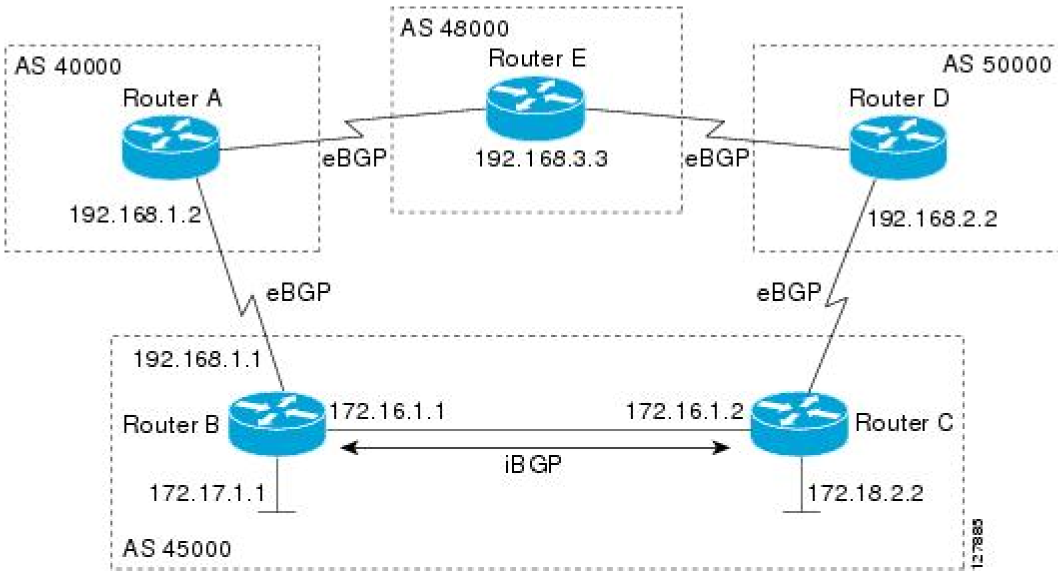
Using the BGP hybrid CLI feature, you can configure commands in the address family VPNv4 format and save these command configurations without modifying an existing NLRI formatted configuration. If you want to use other address family configurations such as IPv4 unicast or multicast, then you must upgrade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

SAFI Field Value	Description	Reference
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.

Where to Go Next

Proceed to 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 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 http://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 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.

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.
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.

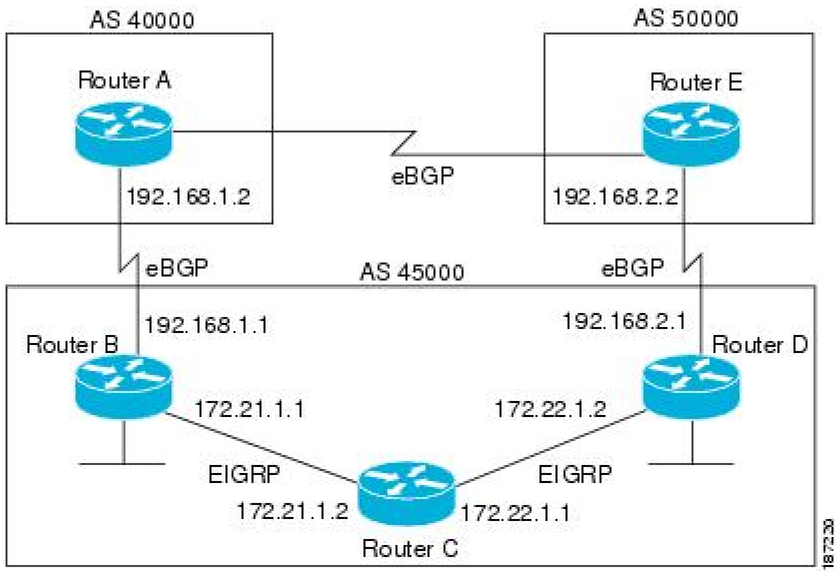
BGP Peer Groups

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.

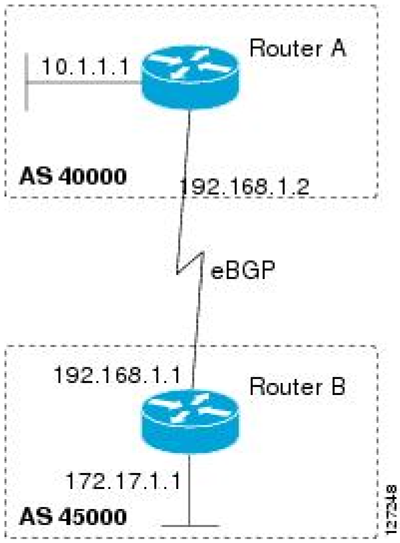
Configuring a BGP Routing Process

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.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:**	Configures a BGP routing process, and enters router configuration mode for the specified routing process.

	Command or Action	Purpose
	`Device(config)# router bgp 40000`	• Use the *autonomous-system-number* argument to specify an integer, from 0 and 65534, that identifies the device to other BGP speakers.
Step 4	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router)# network 10.1.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols, the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 5	**bgp router-id** *ip-address* **Example:** `Device(config-router)# bgp router-id 10.1.1.99`	(Optional) Configures a fixed 32-bit router ID as the identifier of the local device running BGP. • Use the *ip-address* argument to specify a unique router ID within the network. **Note** Configuring a router ID using the **bgp router-id** command resets all active BGP peering sessions.
Step 6	**timers bgp** *keepalive holdtime* **Example:** `Device(config-router)# timers bgp 70 120`	(Optional) Sets BGP network timers. • Use the *keepalive* argument to specify the frequency, in seconds, with which the software sends keepalive messages to its BGP peer. By default, the keepalive timer is set to 60 seconds. • Use the *holdtime* argument to specify the interval, in seconds, after which the software, having not received a keepalive message, declares a BGP peer dead. By default, the holdtime timer is set to 180 seconds.
Step 7	**bgp fast-external-fallover** **Example:** `Device(config-router)# bgp fast-external-fallover`	(Optional) Enables the automatic resetting of BGP sessions. • By default, the BGP sessions of any directly adjacent external peers are reset if the link used to reach them goes down.
Step 8	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	(Optional) Enables logging of BGP neighbor status changes (up or down) and neighbor resets. • Use this command for troubleshooting network connectivity problems and measuring network stability. Unexpected neighbor resets might indicate high error rates or high packet loss in the network and should be investigated.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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
```

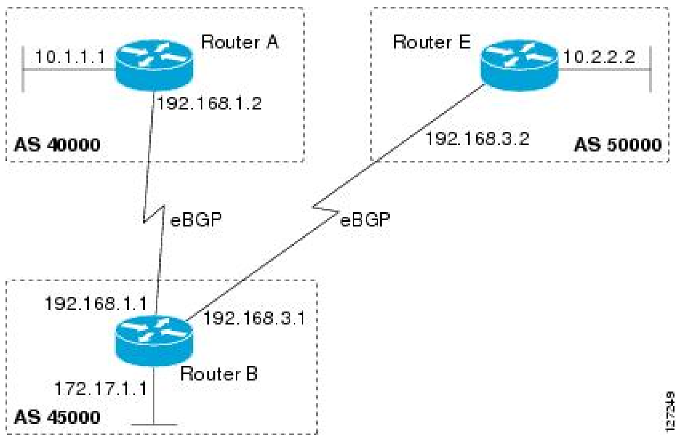
Troubleshooting Tips

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

	Command or Action	Purpose
		• Use the **export** keyword to export routing information to the target VPN extended community. • Use the **both** keyword to import both import and export routing information to the target VPN extended community. • Use the *route-target-ext-community* argument to add the route target extended community attributes to the VRF's list of import, export, or both (import and export) route target extended communities.
Step 10	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 11	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 12	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 vrf vpn1`	Specifies the IPv4 address family and enters address family configuration mode. • Use the **unicast** keyword to specify the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • Use the **multicast** keyword to specify IPv4 multicast address prefixes. • Use the **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.3.2 remote-as 50000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 14	**neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**] **Example:**	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

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 192.168.3.2 maximum-prefix 10000 warning-only`	neighbor. The number of prefixes that can be configured is limited only by the available system resources on a device. • Use the *threshold* argument to specify an integer representing a percentage of the maximum prefix limit at which the device starts to generate a warning message. • Use the **warning-only** keyword to allow the device to generate a log message when the maximum prefix limit is exceeded, instead of terminating the peering session.
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local device.
Step 16	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

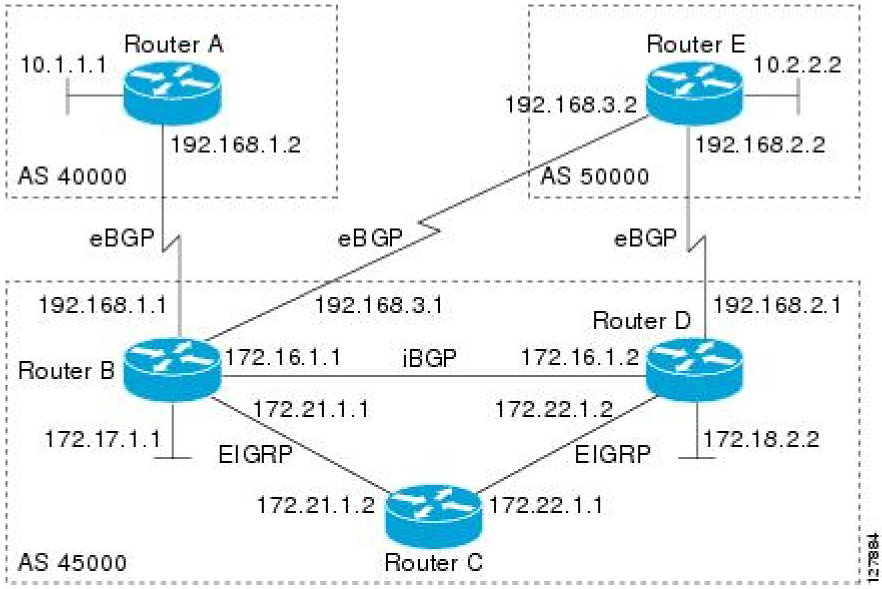
Use the **ping vrf** command to verify basic network connectivity between the BGP devices, and use the **show ip vrf** command to verify that the VRF instance has been created.

Customizing a BGP Peer

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

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

Figure 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. **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

	Command or Action	Purpose
		family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 8	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.2 activate`	Enables the exchange of information with a BGP neighbor.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds* **Example:** `Device(config-router-af)# neighbor 192.168.3.2 advertisement-interval 25`	(Optional) Sets the minimum interval between the sending of BGP routing updates.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*] **Example:** `Device(config-router-af)# neighbor 192.168.3.2 default-originate`	(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.
Step 12	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters router configuration mode.
Step 13	**neighbor** {*ip-address* | *peer-group-name*} **shutdown** **Example:** `Device(config-router)# neighbor 192.168.3.2 shutdown`	(Optional) Disables a BGP peer or peer group. **Note** If you perform this step you will not be able to run either of the subsequent **show** command steps because you have disabled the neighbor.

	Command or Action	Purpose
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 15	**show ip bgp ipv4 multicast** [*command*] **Example:** `Device# show ip bgp ipv4 multicast`	(Optional) Displays IPv4 multicast database-related information. • Use the *command* argument to specify any multiprotocol BGP command that is supported. To see the supported commands, use the ? prompt on the CLI.
Step 16	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received prefix-filter**] **Example:** `Device# show ip bgp neighbors 192.168.3.2`	(Optional) Displays information about the TCP and BGP connections to neighbors.

Examples

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

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

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

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Description: finance
  BGP version 4, remote router ID 10.2.2.99
  BGP state = Established, up for 01:48:27
  Last read 00:00:26, last write 00:00:26, hold time is 120, keepalive intervals
  Configured hold time is 120,keepalive interval is 70 seconds, Minimum holdtims
  Neighbor capabilities:
    Route refresh: advertised and received (old & new)
    Address family IPv4 Unicast: advertised
    Address family IPv4 Multicast: advertised and received
!
```

```
 For address family: IPv4 Multicast
  BGP table version 3, neighbor version 3/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
    Uses NEXT_HOP attribute for MBGP NLRIs
                                Sent       Rcvd
  Prefix activity:              ----       ----
    Prefixes Current:              1          1 (Consumes 48 bytes)
    Prefixes Total:                1          1
    Implicit Withdraw:             0          0
    Explicit Withdraw:             0          0
    Used as bestpath:            n/a          1
    Used as multipath:           n/a          0
                                  Outbound    Inbound
  Local Policy Denied Prefixes:    --------    -------
    Bestpath from this peer:              1        n/a
    Total:                                1          0
  Number of NLRIs in the update sent: max 0, min 0
  Minimum time between advertisement runs is 25 seconds
  Connections established 8; dropped 7
  Last reset 01:48:54, due to User reset
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 192.168.3.1, Local port: 13172
Foreign host: 192.168.3.2, Foreign port: 179
!
```

Removing BGP Configuration Commands Using a Redistribution

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

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

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

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

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

	Command or Action	Purpose
	`Device# show running-config`	• 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.

	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	**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.

	Command or Action	Purpose
Step 9	Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.	—
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LOCAL permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named LOCAL is created.
Step 12	**set ip next-hop** *ip-address* **Example:** `Device(config-route-map)# set ip next-hop 192.168.1.144`	Specifies where output packets that pass a match clause of a route map for policy routing. • In this example, the ip address is set to 192.168.1.144.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 15	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

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

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

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

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

Resetting and Displaying Basic BGP Information

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

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:**	Clears and resets BGP neighbor sessions: • In the example provided, all BGP neighbor sessions are cleared and reset.

	Command or Action	Purpose
	`Device# clear ip bgp *`	
Step 3	**show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes** *mask-length*] **Example:** `Device# show ip bgp 10.1.1.0 255.255.255.0`	Displays all the entries in the BGP routing table: • In the example provided, the BGP routing table information for the 10.1.1.0 network is displayed.
Step 4	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*] **Example:** `Device# show ip bgp neighbors 192.168.3.2 advertised-routes`	Displays information about the TCP and BGP connections to neighbors. • In the example provided, the routes advertised from the device to BGP neighbor 192.168.3.2 on another device are displayed.
Step 5	**show ip bgp paths** **Example:** `Device# show ip bgp paths`	Displays information about all the BGP paths in the database.
Step 6	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays information about the status of all BGP connections.

Aggregating Route Prefixes Using BGP

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

Redistributing a Static Aggregate Route into BGP

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

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 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*.

	Command or Action	Purpose
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.
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:**	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
	`Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	

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** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.

	Command or Action	Purpose
Step 5	**neighbor** *ip-address* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*} **Example:** `Device(config-router)# neighbor 192.168.1.2 advertise-map map1 exist-map map2`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device. • In this example, the prefix (172.17.0.0) matching the ACL in the advertise map (the route map named map1) will be advertised to the neighbor only when a prefix (192.168.50.0) matching the ACL in exist map (the route-map named map2) is in the local BGP table.
Step 6	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 7	**route-map** *map-tag* [**permit** | **deny**] [**sequence-number**] **Example:** `Device(config)# route-map map1 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named map1 is created.
Step 8	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# match ip address 1`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by access list 1.
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 10	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map map2 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named map2 is created.
Step 11	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# match ip address 2`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by access list 2.

	Command or Action	Purpose
Step 12	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 13	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 1 permit 172.17.0.0`	Configures a standard access list. • In this example, access list 1 permits advertising of the 172.17.0.0 prefix, depending on other conditions set by the **neighbor advertise-map** command.
Step 14	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 2 permit 192.168.50.0`	Configures a standard access list. • In this example, access list 2 permits the 192.168.50.0 to be the prefix of the exist-map.
Step 15	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Originating BGP Routes

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

Advertising a Default Route Using BGP

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

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

	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.
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* **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.
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.

	Command or Action	Purpose
Step 7	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 multicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. This is the default. • The **multicast** keyword specifies that IPv4 multicast address prefixes will be exchanged. • The **vrf** keyword and *vrf-name* argument specify that IPv4 VRF instance information will be exchanged.
Step 8	**neighbor** *peer-group-name* **activate** **Example:** `Device(config-router-af)# neighbor fingroup activate`	Enables the neighbor to exchange prefixes for the IPv4 address family with the local device. **Note** By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only unicast address prefixes. To allow BGP to exchange other address prefix types, such as multicast that is configured in this example, neighbors must also be activated using the **neighbor activate** command.
Step 9	**neighbor** *ip-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 peer-group fingroup`	Assigns the IP address of a BGP neighbor to a peer group.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP 4

Example: Configuring a BGP Process and Customizing Peers

The following example shows the configuration for Router B in the 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*

Standard/RFC	Title
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 http://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	11.2(1)	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).

CHAPTER 3

Configuring a Basic BGP Network

This module describes the basic tasks to configure a basic Border Gateway Protocol (BGP) network. BGP is an interdomain routing protocol that is designed to provide loop-free routing between organizations. The Cisco IOS implementation of the neighbor and address family commands is explained. This module also contains tasks to configure and customize BGP peers, implement BGP route aggregation, configure BGP route origination, and define BGP backdoor routes. BGP peer group definition is documented, peer session templates are introduced, and update groups are explained,

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 Configuring a Basic BGP Network

Before configuring a basic BGP network, you should be familiar with the "Cisco BGP Overview" module.

Restrictions for Configuring a Basic BGP Network

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 address family configurations.

Information About Configuring a Basic BGP Network

Cisco Implementation of BGP Global and Address Family Configuration Commands

The address family model for configuring BGP is based on splitting apart the configuration for each address family. All commands that are independent of the address family are grouped together at the beginning (highest level) of the configuration, and these are followed by separate submodes for commands specific to each address family (with the exception that commands relating to IPv4 unicast can also be entered at the beginning of the configuration). When a network operator configures BGP, the flow of BGP configuration categories is represented by the following bullets in order:

- Global configuration—Configuration that is applied to BGP in general, rather than to specific neighbors. For example, the **network**, **redistribute**, and **bgp bestpath** commands.
- Address family-dependent configuration—Configuration that applies to a specific address family such as policy on an individual neighbor.

The relationship between BGP global and BGP address family-dependent configuration categories is shown in the table below.

Table 8: Relationships Between BGP Configuration Categories

BGP Configuration Category	Configuration Sets Within Category
Global address family-independent	One set of global address family-independent configurations
Address family-dependent	One set of global address family-dependent configurations per address family

Note Address family configuration must be entered within the address family submode to which it applies.

The following is an example of BGP configuration statements showing the grouping of global address family-independent and address family-dependent commands.

```
router bgp <AS>
 ! AF independent part
 neighbor <ip-address> <command> ! Session config; AF independent
 address-family ipv4 unicast
  ! AF dependant part
```

```
  neighbor <ip-address> <command> ! Policy config; AF dependant
  exit-address-family
 address-family ipv4 multicast
  ! AF dependant part
  neighbor <ip-address> <command> ! Policy config; AF dependant
  exit-address-family
 address-family ipv4 unicast vrf <vrf-name>
  ! VRF specific AS independent commands
  ! VRF specific AS dependant commands
  neighbor <ip-address> <command> ! Session config; AF independent
  neighbor <ip-address> <command> ! Policy config; AF dependant
  exit-address-family
```

The following example shows actual BGP commands that match the BGP configuration statements in the previous example:

```
router bgp 45000
 router-id 172.17.1.99
 bgp log-neighbor-changes
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 address-family ipv4 unicast
  neighbor 192.168.1.2 activate
  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
  network 172.16.1.0 mask 255.255.255.0
  exit-address-family
 address-family ipv4 vrf vpn1
  neighbor 192.168.3.2 activate
  network 172.21.1.0 mask 255.255.255.0
  exit-address-family
```

The **bgp upgrade-cli** command simplifies the migration of BGP networks and existing configurations from the network layer reachability information (NLRI) format to the address family format. Network operators can configure commands in the address family identifier (AFI) format and save these command configurations to existing NLRI formatted configurations. The BGP hybrid command-line interface (CLI) does not add support for complete AFI and NLRI integration because of the limitations of the NLRI format. For complete support of AFI commands and features, we recommend upgrading existing NLRI configurations with the **bgp upgrade-cli** command. For an example of migrating BGP configurations from the NLRI format to the address family format, see the “Example: NLFI to AFI Configuration” section later in this module.

Conditional BGP Route Injection

Routes that are advertised through the BGP are commonly aggregated to minimize the number of routes that are used and reduce the size of global routing tables. However, common route aggregation can obscure more specific routing information that is more accurate but not necessary to forward packets to their destinations. Routing accuracy is obscured by common route aggregation because a prefix that represents multiple addresses or hosts over a large topological area cannot be accurately reflected in a single route. Cisco software provides several methods by which you can originate a prefix into BGP. Prior to the BGP conditional route injection feature, the existing methods included redistribution and using the **network** or **aggregate-address** command. However, these methods assume the existence of more specific routing information (matching the route to be originated) in either the routing table or the BGP table.

BGP conditional route injection allows you to originate a prefix into a BGP routing table without the corresponding match. This feature allows more specific routes to be generated based on administrative policy or traffic engineering information in order to provide more specific control over the forwarding of packets to these more specific routes, which are injected into the BGP routing table only if the configured conditions are met. Enabling this feature will allow you to improve the accuracy of common route aggregation by conditionally injecting or replacing less specific prefixes with more specific prefixes. Only prefixes that are equal to or more specific than the original prefix may be injected. BGP conditional route injection is enabled with the **bgp inject-map exist-map**command and uses two route maps (inject map and exist map) to install one (or more) more specific prefixes into a BGP routing table. The exist map specifies the prefixes that the BGP speaker will track. The inject map defines the prefixes that will be created and installed into the local BGP table.

Note Inject maps and exist maps will only match a single prefix per route map clause. To inject additional prefixes, you must configure additional route map clauses. If multiple prefixes are used, the first prefix matched will be used.

BGP Update Group

The introduction of the BGP (dynamic) update group provides a different type of BGP peer grouping from existing BGP peer groups. Existing peer groups are not affected but peers with the same outbound policy configured that are not members of a current peer group can be grouped into an update group. The members of this update group will use the same update generation engine. When BGP update groups are configured an algorithm dynamically calculates the BGP update group membership based on outbound policies. Optimal BGP update message generation occurs automatically and independently. BGP neighbor configuration is no longer restricted by outbound routing policies, and update groups can belong to different address families.

BGP Dynamic Update Group Configuration

In Cisco IOS Release 12.0(24)S, 12.2(18)S, 12.3(4)T, 12.2(27)SBC, and later releases, a new algorithm was introduced that dynamically calculates and optimizes update groups of neighbors that share the same outbound policies and can share the same update messages. No configuration is required to enable the BGP dynamic update group and the algorithm runs automatically. When a change to outbound policy occurs, the router automatically recalculates update group memberships and applies the changes by triggering an outbound soft reset after a 1-minute timer expires. This behavior is designed to provide the network operator with time to change the configuration if a mistake is made. You can manually enable an outbound soft reset before the timer expires by entering the **clear ip bgp** *ip-address* **soft out**command.

Note In Cisco IOS Release 12.0(22)S, 12.2(14)S, 12.3(2)T, and prior releases, the update group recalculation delay timer is set to 3 minutes.

For the best optimization of BGP update group generation, we recommend that the network operator keeps outbound routing policy the same for neighbors that have similar outbound policies.

BGP Peer Templates

To address some of the limitations of peer groups such as configuration management, BGP peer templates were introduced to support the BGP update group configuration.

A peer template is a configuration pattern that can be applied to neighbors that share policies. Peer templates are reusable and support inheritance, which allows the network operator to group and apply distinct neighbor configurations for BGP neighbors that share policies. Peer templates also allow the network operator to define very complex configuration patterns through the capability of a peer template to inherit a configuration from another peer template.

There are two types of peer templates:

- Peer session templates are used to group and apply the configuration of general session commands that are common to all address family and NLRI configuration modes.
- Peer policy templates are used to group and apply the configuration of commands that are applied within specific address families and NLRI configuration modes.

Peer templates improve the flexibility and enhance the capability of neighbor configuration. Peer templates also provide an alternative to peer group configuration and overcome some limitations of peer groups. BGP peer routers using peer templates also benefit from automatic update group configuration. With the configuration of the BGP peer templates and the support of the BGP dynamic update peer groups, the network operator no longer needs to configure peer groups in BGP and the network can benefit from improved configuration flexibility and faster convergence.

Note A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies from peer templates.

The following restrictions apply to the peer policy templates:

- A peer policy template can directly or indirectly inherit up to eight peer policy templates.
- A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies only from peer templates.

Inheritance in Peer Templates

The inheritance capability is a key component of peer template operation. Inheritance in a peer template is similar to node and tree structures commonly found in general computing, for example, file and directory trees. A peer template can directly or indirectly inherit the configuration from another peer template. The directly inherited peer template represents the tree in the structure. The indirectly inherited peer template represents a node in the tree. Because each node also supports inheritance, branches can be created that apply the configurations of all indirectly inherited peer templates within a chain back to the directly inherited peer template or the source of the tree.

This structure eliminates the need to repeat configuration statements that are commonly reapplied to groups of neighbors because common configuration statements can be applied once and then indirectly inherited by peer templates that are applied to neighbor groups with common configurations. Configuration statements that are duplicated separately within a node and a tree are filtered out at the source of the tree by the directly inherited template. A directly inherited template will overwrite any indirectly inherited statements that are duplicated in the directly inherited template.

Inheritance expands the scalability and flexibility of neighbor configuration by allowing you to chain together peer templates configurations to create simple configurations that inherit common configuration statements or complex configurations that apply very specific configuration statements along with common inherited configurations. Specific details about configuring inheritance in peer session templates and peer policy templates are provided in the following sections.

When BGP neighbors use inherited peer templates it can be difficult to determine which policies are associated with a specific template. The **detail** keyword was added to the **show ip bgp template peer-policy** command to display the detailed configuration of local and inherited policies associated with a specific template.

Peer Session Templates

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

- **description**
- **disable-connected-check**
- **ebgp-multihop**
- **exit peer-session**
- **inherit peer-session**
- **local-as**
- **password**
- **remote-as**
- **shutdown**
- **timers**
- **translate-update**
- **update-source**
- **version**

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

Peer session templates support direct and indirect inheritance. A peer can be configured with only one peer session template at a time, and that peer session template can contain only one indirectly inherited peer session template.

Note If you attempt to configure more than one inherit statement with a single peer session template, an error message will be displayed.

This behavior allows a BGP neighbor to directly inherit only one session template and indirectly inherit up to seven additional peer session templates. This allows you to apply up to a maximum of eight peer session configurations to a neighbor: the configuration from the directly inherited peer session template and the configurations from up to seven indirectly inherited peer session templates. Inherited peer session configurations are evaluated first and applied starting with the last node in the branch and ending with the directly applied peer session template configuration at the source of the tree. The directly applied peer session template will have priority over inherited peer session template configurations. Any configuration statements that are duplicated in inherited peer session templates will be overwritten by the directly applied peer session template. So, if a general session command is reapplied with a different value, the subsequent value will have priority and overwrite the previous value that was configured in the indirectly inherited template. The following examples illustrate the use of this feature.

In the following example, the general session command **remote-as 1** is applied in the peer session template named SESSION-TEMPLATE-ONE:

```
template peer-session SESSION-TEMPLATE-ONE
 remote-as 1
 exit peer-session
```

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

Peer Policy Templates

Peer policy templates are used to group and apply the configuration of commands that are applied within specific address families and NLRI configuration mode. Peer policy templates are created and configured in peer policy configuration mode. BGP policy commands that are configured for specific address families are configured in a peer policy template. The following BGP policy commands are supported by peer policy templates:

- **advertisement-interval**
- **allowas-in**
- **as-override**
- **capability**
- **default-originate**
- **distribute-list**
- **dmzlink-bw**
- **exit-peer-policy**
- **filter-list**
- **inherit peer-policy**
- **maximum-prefix**
- **next-hop-self**
- **next-hop-unchanged**

- **prefix-list**
- **remove-private-as**
- **route-map**
- **route-reflector-client**
- **send-community**
- **send-label**
- **soft-reconfiguration**
- **unsuppress-map**
- **weight**

Peer policy templates are used to configure BGP policy commands that are configured for neighbors that belong to specific address families. Like peer session templates, peer policy templates are configured once and then applied to many neighbors through the direct application of a peer policy template or through inheritance from peer policy templates. The configuration of peer policy templates simplifies the configuration of BGP policy commands that are applied to all neighbors within an autonomous system.

Like a peer session template, a peer policy template supports inheritance. However, there are minor differences. A directly applied peer policy template can directly or indirectly inherit configurations from up to seven peer policy templates. So, a total of eight peer policy templates can be applied to a neighbor or neighbor group. Like route maps, inherited peer policy templates are configured with sequence numbers. Also like a route map, an inherited peer policy template is evaluated starting with the **inherit peer-policy** statement with the lowest sequence number and ending with the highest sequence number. However, there is a difference; a peer policy template will not collapse like a route map. Every sequence is evaluated, and if a BGP policy command is reapplied with a different value, it will overwrite any previous value from a lower sequence number.

The directly applied peer policy template and the **inherit peer-policy** statement with the highest sequence number will always have priority and be applied last. Commands that are reapplied in subsequent peer templates will always overwrite the previous values. This behavior is designed to allow you to apply common policy configurations to large neighbor groups and specific policy configurations only to certain neighbors and neighbor groups without duplicating individual policy configuration commands.

Peer policy templates support only policy configuration commands. BGP policy configuration commands that are configured only for specific address families are configured with peer policy templates.

The configuration of peer policy templates simplifies and improves the flexibility of BGP configuration. A specific policy can be configured once and referenced many times. Because a peer policy supports up to eight levels of inheritance, very specific and very complex BGP policies can also be created.

BGP IPv6 Neighbor Activation Under the IPv4 Address Family

Prior to Cisco IOS Release 12.2(33)SRE4, by default, both IPv6 and IPv4 capability is exchanged with a BGP peer that has an IPv6 address. When an IPv6 peer is configured, that neighbor is automatically activated under the IPv4 unicast address family.

Beginning with Cisco IOS Release 12.2(33)SRE4, when a *new* IPv6 neighbor is being configured, it is no longer automatically activated under the IPv4 address family. You can manually activate the IPv6 neighbor under the IPv4 address family if, for example, you have a dual stack environment and want to send IPv6 and IPv4 prefixes.

If you do not want an *existing* IPv6 peer to be activated under the IPv4 address family, you can manually deactivate the peer with the **no neighbor activate** command. Until then, existing configurations that activate an IPv6 neighbor under the IPv4 unicast address family will continue to try to establish a session.

How to Configure a Basic BGP Network

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. The other tasks in the following list are optional:

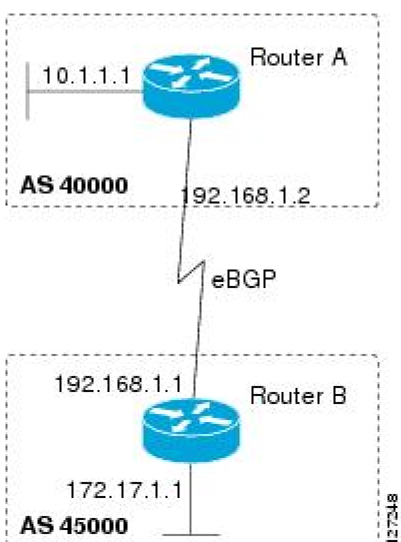
Configuring a BGP Routing Process

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 9: 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.
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.

	Command or Action	Purpose
Step 6	**timers bgp** *keepalive holdtime* **Example:** `Device(config-router)# timers bgp 70 120`	(Optional) Sets BGP network timers. • Use the *keepalive* argument to specify the frequency, in seconds, with which the software sends keepalive messages to its BGP peer. By default, the keepalive timer is set to 60 seconds. • Use the *holdtime* argument to specify the interval, in seconds, after which the software, having not received a keepalive message, declares a BGP peer dead. By default, the holdtime timer is set to 180 seconds.
Step 7	**bgp fast-external-fallover** **Example:** `Device(config-router)# bgp fast-external-fallover`	(Optional) Enables the automatic resetting of BGP sessions. • By default, the BGP sessions of any directly adjacent external peers are reset if the link used to reach them goes down.
Step 8	**bgp log-neighbor-changes** **Example:** `Device(config-router)# bgp log-neighbor-changes`	(Optional) Enables logging of BGP neighbor status changes (up or down) and neighbor resets. • Use this command for troubleshooting network connectivity problems and measuring network stability. Unexpected neighbor resets might indicate high error rates or high packet loss in the network and should be investigated.
Step 9	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 10	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

Troubleshooting Tips

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

Configuring a BGP Peer

Perform this task to configure BGP between two IPv4 routers (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 neighbor routers that are to be BGP peers.

Before you begin

Before you perform this task, perform the “Configuring a BGP Routing Process” task shown in the prior section.

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

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(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 router.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 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:** `Router(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local router.
Step 7	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.
Step 8	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# 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:**	(Optional) Displays information about the TCP and BGP connections to neighbors.

	Command or Action	Purpose
	`Router(config-router-af)# show ip bgp neighbors 192.168.2.2`	**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.

What to Do Next

If you have BGP peers in a VPN, proceed to the Configuring a BGP Peer for the IPv4 VRF Address Family, on page 88. If you do not have BGP peers in a VPN, proceed to the Customizing a BGP Peer, on page 35.

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

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

Before you begin

Note

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Device(config-router)# neighbor 192.168.1.2 remote-as 65536`	
Step 5	Repeat Step 4 to define other BGP neighbors, as required.	--
Step 6	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.
Step 7	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local device.
Step 8	Repeat Step 7 to activate other BGP neighbors, as required.	--
Step 9	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this AS and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 11	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp 10.1.1.0`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

	Command or Action	Purpose
Step 12	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	(Optional) Displays the status of all BGP connections.

Examples

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

```
RouterB# show ip bgp 10.1.1.0

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

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

```
RouterB# show ip bgp summary

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

Troubleshooting Tips

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays the status of all Border Gateway Protocol (BGP) connections.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 5	**bgp asnotation dot** **Example:** `Device(config-router)# bgp asnotation dot`	Changes the default output format of BGP 4-byte AS numbers from asplain (decimal values) to dot notation. **Note** 4-byte AS numbers can be configured using either asplain format or asdot format. This command affects only the output displayed for **show** commands or the matching of regular expressions.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config-router)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 7	**clear ip bgp *** **Example:** `Device# clear ip bgp *`	Clears and resets all current BGP sessions. • In this example, a hard reset is performed to ensure that the 4-byte AS number format change is reflected in all BGP sessions. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 8	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays the status of all BGP connections.
Step 9	**show ip bgp regexp** *regexp* **Example:** `Device# show ip bgp regexp ^1\.0$`	Displays routes that match the AS path regular expression. • In this example, a regular expression to match a 4-byte AS path is configured using asdot format.
Step 10	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 11	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 12	**no bgp asnotation dot** **Example:** `Device(config-router)# no bgp asnotation dot`	Resets the default output format of BGP 4-byte AS numbers back to asplain (decimal values). **Note** 4-byte AS numbers can be configured using either asplain format or asdot format. This command affects only the output displayed for **show** commands or the matching of regular expressions.
Step 13	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

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

Examples

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

```
Router# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 65538
BGP table version is 1, main routing table version 1
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  Statd
192.168.1.2     4       65536       7       7        1    0    0 00:03:04      0
192.168.3.2     4       65550       4       4        1    0    0 00:00:15      0
```

After the **bgp asnotation dot** command is configured (followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions), the output is converted to asdot notation format as shown in the following output from the **show ip bgp summary** command. Note the asdot format of the 4-byte AS numbers, 1.0 and 1.14 (these are the asdot conversions of the 65536 and 65550 AS numbers.

```
Router# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 1.2
BGP table version is 1, main routing table version 1
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  Statd
192.168.1.2     4          1.0       9       9        1    0    0 00:04:13      0
192.168.3.2     4         1.14       6       6        1    0    0 00:01:24      0
```

After the **bgp asnotation dot** command is configured (followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions), the regular expression match format for 4-byte AS paths is changed to asdot notation format. Although a 4-byte AS number can be configured in a regular expression using either asplain format or asdot format, only 4-byte AS numbers configured using the current default format are matched. In the first example below, the **show ip bgp regexp** command is configured with a 4-byte AS number in asplain format. The match fails because the default format is currently asdot format and there is no output. In the second example using asdot format, the match passes and the information about the 4-byte AS path is shown using the asdot notation.

Note The asdot notation uses a period, which is a special character in Cisco regular expressions. To remove the special meaning, use a backslash before the period.

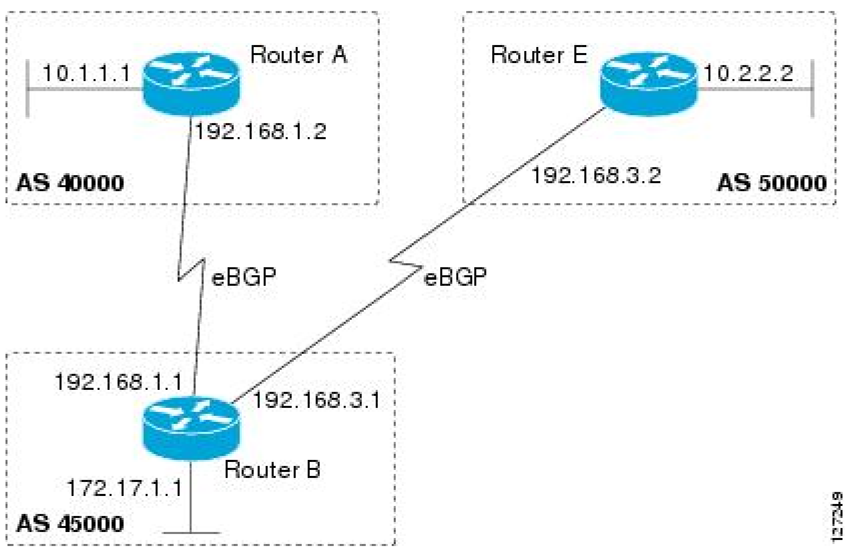
```
Router# show ip bgp regexp ^65536$

Router# show ip bgp regexp ^1\.0$

BGP table version is 2, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 1.0 i
```

Configuring a BGP Peer for the IPv4 VRF Address Family

Perform this optional task to configure BGP between two IPv4 routers (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 neighbor routers that are to be BGP IPv4 VRF address family peers.

This task does not show the complete configuration required for VPN routing. For some complete example configurations and an example configuration showing how to create a VRF with a route-target that uses a 4-byte autonomous system number, see .

Figure 10: BGP Topology for IPv4 VRF Address Family



Before you begin

Before you perform this task, perform the Configuring a BGP Routing Process, on page 25 task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*

5. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
6. **exit**
7. **router bgp** *autonomous-system-number*
8. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
9. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
10. **neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**]
11. **neighbor** *ip-address* **activate**
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Router(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 4	**rd** *route-distinguisher* **Example:** `Router(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 5	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Router(config-vrf)# route-target both 45000:100`	Creates a route target extended community for a VRF. • Use the **import** keyword to import routing information from the target VPN extended community. • Use the **export** keyword to export routing information to the target VPN extended community. • Use the **both** keyword to 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

	Command or Action	Purpose
		VRF's list of import, export, or both (import and export) route target extended communities.
Step 6	**exit** **Example:** `Router(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 7	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 8	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 router is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • 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 9	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.3.2 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **maximum-prefix** *maximum* [*threshold*] [**restart** *restart-interval*] [**warning-only**] **Example:** `Router(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 router. • Use the *threshold* argument to specify an integer representing a percentage of the maximum prefix limit at which the router starts to generate a warning message.

	Command or Action	Purpose
		• Use the **warning-only** keyword to allow the router to generate a log message when the maximum prefix limit is exceeded, instead of terminating the peering session.
Step 11	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af)# neighbor 192.168.3.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local router.
Step 12	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

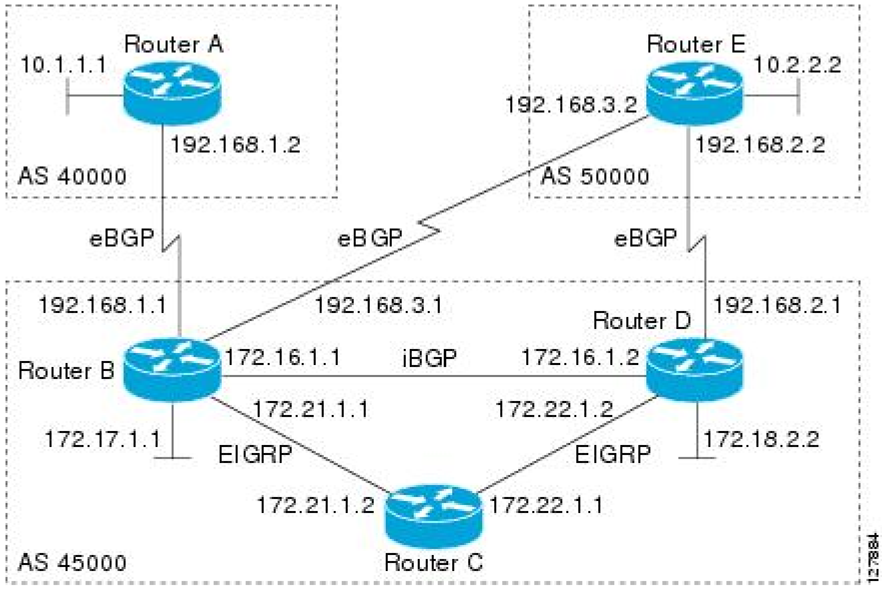
Use the **ping** command to verify basic network connectivity between the BGP routers, and use the **show ip vrf** command to verify that the VRF instance has been created.

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 11: 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. **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

	Command or Action	Purpose
		family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 8	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.3.2 activate`	Enables the exchange of information with a BGP neighbor.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **advertisement-interval** *seconds* **Example:** `Device(config-router-af)# neighbor 192.168.3.2 advertisement-interval 25`	(Optional) Sets the minimum interval between the sending of BGP routing updates.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*] **Example:** `Device(config-router-af)# neighbor 192.168.3.2 default-originate`	(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.
Step 12	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters router configuration mode.
Step 13	**neighbor** {*ip-address* | *peer-group-name*} **shutdown** **Example:** `Device(config-router)# neighbor 192.168.3.2 shutdown`	(Optional) Disables a BGP peer or peer group. **Note** If you perform this step you will not be able to run either of the subsequent **show** command steps because you have disabled the neighbor.

	Command or Action	Purpose
Step 14	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 15	**show ip bgp ipv4 multicast** [*command*] **Example:** `Device# show ip bgp ipv4 multicast`	(Optional) Displays IPv4 multicast database-related information. • Use the *command* argument to specify any multiprotocol BGP command that is supported. To see the supported commands, use the ? prompt on the CLI.
Step 16	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received prefix-filter**] **Example:** `Device# show ip bgp neighbors 192.168.3.2`	(Optional) Displays information about the TCP and BGP connections to neighbors.

Examples

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

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

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

```
BGP neighbor is 192.168.3.2,  remote AS 50000, external link
 Description: finance
  BGP version 4, remote router ID 10.2.2.99
  BGP state = Established, up for 01:48:27
  Last read 00:00:26, last write 00:00:26, hold time is 120, keepalive intervals
  Configured hold time is 120,keepalive interval is 70 seconds, Minimum holdtims
  Neighbor capabilities:
    Route refresh: advertised and received (old & new)
    Address family IPv4 Unicast: advertised
    Address family IPv4 Multicast: advertised and received
!
```

```
 For address family: IPv4 Multicast
  BGP table version 3, neighbor version 3/0
 Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
    Uses NEXT_HOP attribute for MBGP NLRIs
                              Sent       Rcvd
  Prefix activity:            ----       ----
    Prefixes Current:            1          1 (Consumes 48 bytes)
    Prefixes Total:              1          1
    Implicit Withdraw:           0          0
    Explicit Withdraw:           0          0
    Used as bestpath:          n/a          1
    Used as multipath:         n/a          0
                                Outbound    Inbound
  Local Policy Denied Prefixes:    --------    -------
    Bestpath from this peer:              1        n/a
    Total:                                1          0
  Number of NLRIs in the update sent: max 0, min 0
  Minimum time between advertisement runs is 25 seconds
  Connections established 8; dropped 7
  Last reset 01:48:54, due to User reset
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Local host: 192.168.3.1, Local port: 13172
Foreign host: 192.168.3.2, Foreign port: 179
!
```

Removing BGP Configuration Commands Using a Redistribution

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

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

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

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

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

	Command or Action	Purpose
	`Device# show running-config`	• 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 routers 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.

	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	**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.

	Command or Action	Purpose
Step 9	Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.	—
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LOCAL permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named LOCAL is created.
Step 12	**set ip next-hop** *ip-address* **Example:** `Device(config-route-map)# set ip next-hop 192.168.1.144`	Specifies where output packets that pass a match clause of a route map for policy routing. • In this example, the ip address is set to 192.168.1.144.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 15	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

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

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

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

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

Resetting and Displaying Basic BGP Information

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

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:**	Clears and resets BGP neighbor sessions: • In the example provided, all BGP neighbor sessions are cleared and reset.

	Command or Action	Purpose
	`Device# clear ip bgp *`	
Step 3	**show ip bgp** [*network-address*] [*network-mask*] [**longer-prefixes**] [**prefix-list** *prefix-list-name* | **route-map** *route-map-name*] [**shorter prefixes** *mask-length*] **Example:** `Device# show ip bgp 10.1.1.0 255.255.255.0`	Displays all the entries in the BGP routing table: • In the example provided, the BGP routing table information for the 10.1.1.0 network is displayed.
Step 4	**show ip bgp neighbors** [*neighbor-address*] [**received-routes** | **routes** | **advertised-routes** | **paths** *regexp* | **dampened-routes** | **received** *prefix-filter*] **Example:** `Device# show ip bgp neighbors 192.168.3.2 advertised-routes`	Displays information about the TCP and BGP connections to neighbors. • In the example provided, the routes advertised from the device to BGP neighbor 192.168.3.2 on another device are displayed.
Step 5	**show ip bgp paths** **Example:** `Device# show ip bgp paths`	Displays information about all the BGP paths in the database.
Step 6	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays information about the status of all BGP connections.

Aggregating Route Prefixes Using BGP

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

Redistributing a Static Aggregate Route into BGP

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

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 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*.

	Command or Action	Purpose
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.
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:**	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
	`Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	

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.

Suppressing Inactive Route Advertisement Using BGP

Perform this task to suppress the advertisement of inactive routes by BGP. In Cisco IOS Release 12.2(25)S, 12.2(33)SXH, and 15.0(1)M, the **bgp suppress-inactive** command was introduced to configure BGP to not advertise inactive routes to any BGP peer. A BGP routing process can advertise routes that are not installed in the 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. This feature can be configured on a per IPv4 address family basis. For example, when specifying the maximum number of routes that can be configured in a VRF with the **maximum routes** global configuration command, you also suppress inactive route advertisement to prevent inactive routes from being accepted into the VRF after route limit has been exceeded.

Before you begin

This task assumes that BGP is enabled and that peering has been established.

Note Inactive route suppression can be configured only under the IPv4 address family or under a default IPv4 general session.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
5. **bgp suppress-inactive**
6. **end**
7. **show ip bgp rib-failure**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]} **Example:** `Router(config-router)# address-family ipv4 unicast`	Enter address family configuration mode to configure BGP peers to accept address family specific configurations. • The example creates an IPv4 unicast address family session.
Step 5	**bgp suppress-inactive** **Example:**	Suppresses BGP advertising of inactive routes. • BGP advertises inactive routes by default.

	Command or Action	Purpose
	`Router(config-router-af)# bgp suppress-inactive`	• Entering the **no** form of this command reenables the advertisement of inactive routes.
Step 6	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.
Step 7	**show ip bgp rib-failure** **Example:** `Router# show ip bgp rib-failure`	(Optional) Displays BGP routes that are not installed in the RIB.

Examples

The following example shows output from the **show ip bgp rib-failure** command displaying routes that are not installed in the RIB. The output shows that the displayed routes were not installed because a route or routes with a better administrative distance already exist in the RIB.

```
Router# show ip bgp rib-failure

Network            Next Hop                      RIB-failure   RIB-NH Matches
10.1.15.0/24       10.1.35.5           Higher admin distance              n/a
10.1.16.0/24       10.1.15.1           Higher admin distance              n/a
```

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** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device.
Step 5	**neighbor** *ip-address* **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*} **Example:** `Device(config-router)# neighbor 192.168.1.2 advertise-map map1 exist-map map2`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local device. • In this example, the prefix (172.17.0.0) matching the ACL in the advertise map (the route map named map1) will be advertised to the neighbor only when a prefix (192.168.50.0) matching the ACL in exist map (the route-map named map2) is in the local BGP table.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 7	**route-map** *map-tag* [**permit** | **deny**] [**sequence-number**] **Example:** `Device(config)# route-map map1 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named map1 is created.
Step 8	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# match ip address 1`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by access list 1.
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 10	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map map2 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named map2 is created.
Step 11	**match ip address** {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...* | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]} **Example:** `Device(config-route-map)# match ip address 2`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by access list 2.
Step 12	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 13	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 1 permit 172.17.0.0`	Configures a standard access list. • In this example, access list 1 permits advertising of the 172.17.0.0 prefix, depending on other conditions set by the **neighbor advertise-map** command.

	Command or Action	Purpose
Step 14	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 2 permit 192.168.50.0`	Configures a standard access list. • In this example, access list 2 permits the 192.168.50.0 to be the prefix of the exist-map.
Step 15	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Originating BGP Routes

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

Advertising a Default Route Using BGP

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

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** **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	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network* / *length* | **permit** *network* / *length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list DEFAULT permit 10.1.1.0/24`	Configures an IP prefix list. • In this example, prefix list DEFAULT permits advertising of the 10.1.1.0/24. prefix depending on a match set by the **match ip address** command.
Step 4	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map ROUTE`	Configures a route map and enters route map configuration mode. • In this example, a route map named ROUTE is created.
Step 5	**match ip address** {*access-list-number* [*access-list-number*... | *access-list-name*...] | *access-list-name* [*access-list-number*... | *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name*...]} **Example:** `Device(config-route-map)# match ip address prefix-list DEFAULT`	Configures the route map to match a prefix that is permitted by a standard access list, an extended access list, or a prefix list. • In this example, the route map is configured to match a prefix permitted by prefix list DEFAULT.
Step 6	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 7	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** *map-name*] **Example:** `Device(config-router)# neighbor 192.168.3.2 default-originate`	(Optional) Permits a BGP speaker--the local device--to send the default route 0.0.0.0 to a peer for use as a default route.
Step 9	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Troubleshooting Tips

Use the **show ip route** command on the receiving BGP peer (not on the local router) to verify that the default route has been set. In the output, verify that a line similar to the following showing the default route 0.0.0.0 is present:

```
B*   0.0.0.0/0 [20/0] via 192.168.1.2, 00:03:10
```

Conditionally Injecting BGP Routes

Use this task to inject more specific prefixes into a BGP routing table over less specific prefixes that were selected through normal route aggregation. These more specific prefixes can be used to provide a finer granularity of traffic engineering or administrative control than is possible with aggregated routes. For more information, see the "Conditional BGP Route Injection" section.

Before you begin

This task assumes that the IGP is already configured for the BGP peers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp inject-map** *inject-map-name* **exist-map** *exist-map-name* [**copy-attributes**]
5. **exit**
6. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
7. **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...*]}
8. **match ip route-source** {*access-list-number* | *access-list-name*} [*access-list-number...*| *access-list-name...*]
9. **exit**
10. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
11. **set 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. **set community** {*community-number* [**additive**] [*well-known-community*] | **none**}
13. **exit**
14. **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*]
15. Repeat Step 14 for every prefix list to be created.
16. **exit**
17. **show ip bgp injected-paths**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 4	**bgp inject-map** *inject-map-name* **exist-map** *exist-map-name* [**copy-attributes**] **Example:** `Router(config-router)# bgp inject-map ORIGINATE exist-map LEARNED_PATH`	Specifies the inject map and the exist map for conditional route injection. • Use the **copy-attributes** keyword to specify that the injected route inherit the attributes of the aggregate route.
Step 5	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 6	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map LEARNED_PATH permit 10`	Configures a route map and enters route map configuration mode.
Step 7	**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:** `Router(config-route-map)# match ip address prefix-list SOURCE`	Specifies the aggregate route to which a more specific route will be injected. • In this example, the prefix list named SOURCE is used to redistribute the source of the route.
Step 8	**match ip route-source** {*access-list-number* | *access-list-name*} [*access-list-number...*| *access-list-name...*] **Example:** `Router(config-route-map)# match ip route-source prefix-list ROUTE_SOURCE`	Specifies the match conditions for redistributing the source of the route. • In this example, the prefix list named ROUTE_SOURCE is used to redistribute the source of the route. **Note** The route source is the neighbor address that is configured with the **neighbor remote-as** command. The tracked prefix must come from this neighbor in order for conditional route injection to occur.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Router(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:** `Router(config)# route-map ORIGINATE permit 10`	Configures a route map and enters route map configuration mode.
Step 11	**set 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:** `Router(config-route-map)# set ip address prefix-list ORIGINATED_ROUTES`	Specifies the routes to be injected. • In this example, the prefix list named originated_routes is used to redistribute the source of the route.
Step 12	**set community** {*community-number* [**additive**] [*well-known-community*] | **none**} **Example:** `Router(config-route-map)# set community 14616:555 additive`	Sets the BGP community attribute of the injected route.
Step 13	**exit** **Example:** `Router(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 14	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Router(config)# ip prefix-list SOURCE permit 10.1.1.0/24`	Configures a prefix list. • In this example, the prefix list named SOURCE is configured to permit routes from network 10.1.1.0/24.
Step 15	Repeat Step 14 for every prefix list to be created.	--
Step 16	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 17	**show ip bgp injected-paths** **Example:**	(Optional) Displays information about injected paths.

	Command or Action	Purpose
	`Router# show ip bgp injected-paths`	

Examples

The following sample output is similar to the output that will be displayed when the **show ip bgp injected-paths**command is entered:

```
Router# show ip bgp injected-paths

BGP table version is 11, local router ID is 10.0.0.1
Status codes:s suppressed, d damped, h history, * valid, > best, i -
internal
Origin codes:i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 172.16.0.0       10.0.0.2                              0 ?
*> 172.17.0.0/16    10.0.0.2                              0 ?
```

Troubleshooting Tips

BGP conditional route injection is based on the injection of a more specific prefix into the BGP routing table when a less specific prefix is present. If conditional route injection is not working properly, verify the following:

- If conditional route injection is configured but does not occur, verify the existence of the aggregate prefix in the BGP routing table. The existence (or not) of the tracked prefix in the BGP routing table can be verified with the **show ip bgp**command.
- If the aggregate prefix exists but conditional route injection does not occur, verify that the aggregate prefix is being received from the correct neighbor and the prefix list identifying that neighbor is a /32 match.
- Verify the injection (or not) of the more specific prefix using the **show ip bgp injected-paths**command.
- Verify that the prefix that is being injected is not outside of the scope of the aggregate prefix.
- Ensure that the inject route map is configured with the **set ip address** command and not the **match ip address** command.

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.
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:**	Enters router configuration mode for the specified routing process.

	Command or Action	Purpose
	`Device(config)# router bgp 40000`	
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.

Configuring Peer Session Templates

The following tasks create and configure a peer session template:

Configuring a Basic Peer Session Template

Perform this task to create a basic peer session template with general BGP routing session commands that can be applied to many neighbors using one of the next two tasks.

Note The commands in Step 5 and 6 are optional and could be replaced with any supported general session commands.

Note The following restrictions apply to the peer session templates:

- A peer session template can directly inherit only one session template, and each inherited session template can also contain one indirectly inherited session template. So, a neighbor or neighbor group can be configured with only one directly applied peer session template and seven additional indirectly inherited peer session templates.
- A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies only from peer templates.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-session** *session-template-name*
5. **remote-as** *autonomous-system-number*
6. **timers** *keepalive-interval hold-time*
7. **end**
8. **show ip bgp template peer-session** [*session-template-name*]

DETAILED STEPS

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

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-session** *session-template-name* **Example:** `Router(config-router)# template peer-session INTERNAL-BGP`	Enters session-template configuration mode and creates a peer session template.
Step 5	**remote-as** *autonomous-system-number* **Example:** `Router(config-router-stmp)# remote-as 202`	(Optional) Configures peering with a remote neighbor in the specified autonomous system. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 6	**timers** *keepalive-interval* *hold-time* **Example:** `Router(config-router-stmp)# timers 30 300`	(Optional) Configures BGP keepalive and hold timers. • The hold time must be at least twice the keepalive time. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 7	**end** **Example:** `Router(config-router)# end`	Exits session-template configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp template peer-session** [*session-template-name*] **Example:** `Router# show ip bgp template peer-session`	Displays locally configured peer session templates. • The output can be filtered to display a single peer policy template with the *session-template-name* argument. This command also supports all standard output modifiers.

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 Peer Session Template Inheritance with the inherit peer-session Command

This task configures peer session template inheritance with the **inherit peer-session** command. It creates and configures a peer session template and allows it to inherit a configuration from another peer session template.

Note The commands in Steps 5 and 6 are optional and could be replaced with any supported general session commands.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-session** *session-template-name*
5. **description** *text-string*
6. **update-source** *interface-type interface-number*
7. **inherit peer-session** *session-template-name*
8. **end**
9. **show ip bgp template peer-session** [*session-template-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-session** *session-template-name* **Example:**	Enter session-template configuration mode and creates a peer session template.

	Command or Action	Purpose
	`Router(config-router)# template peer-session CORE1`	
Step 5	**description** *text-string* **Example:** `Router(config-router-stmp)# description CORE-123`	(Optional) Configures a description. • The text string can be up to 80 characters. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 6	**update-source** *interface-type interface-number* **Example:** `Router(config-router-stmp)# update-source loopback 1`	(Optional) Configures a router to select a specific source or interface to receive routing table updates. • The example uses a loopback interface. The advantage to this configuration is that the loopback interface is not as susceptible to the effects of a flapping interface. **Note** Any supported general session command can be used here. For a list of the supported commands, see the "Restrictions" section.
Step 7	**inherit peer-session** *session-template-name* **Example:** `Router(config-router-stmp)# inherit peer-session INTERNAL-BGP`	Configures this peer session template to inherit the configuration of another peer session template. • The example configures this peer session template to inherit the configuration from INTERNAL-BGP. This template can be applied to a neighbor, and the configuration INTERNAL-BGP will be applied indirectly. No additional peer session templates can be directly applied. However, the directly inherited template can contain up to seven indirectly inherited peer session templates.
Step 8	**end** **Example:** `Router(config-router)# end`	Exits session-template configuration mode and enters privileged EXEC mode.
Step 9	**show ip bgp template peer-session** [*session-template-name*] **Example:** `Router# show ip bgp template peer-session`	Displays locally configured peer session templates. • The output can be filtered to display a single peer policy template with the optional *session-template-name* argument. This command also supports all standard output modifiers.

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 Peer Session Template Inheritance with the neighbor inherit peer-session Command

This task configures a router to send a peer session template to a neighbor to inherit the configuration from the specified peer session template with the **neighbor inherit peer-session** command. Use the following steps to send a peer session template configuration to a neighbor to inherit.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. **neighbor** *ip-address* **inherit peer-session** *session-template-name*
6. **end**
7. **show ip bgp template peer-session** [*session-template-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 101`	Enters router configuration mode and creates a BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 172.16.0.1 remote-as 202`	Configures a peering session with the specified neighbor. • The explicit **remote-as** statement is required for the neighbor inherit statement in Step 5 to work. If a peering is not configured, the specified neighbor in Step 5 will not accept the session template.
Step 5	**neighbor** *ip-address* **inherit peer-session** *session-template-name*	Sends a peer session template to a neighbor so that the neighbor can inherit the configuration.

	Command or Action	Purpose
	Example: `Router(config-router)# neighbor 172.16.0.1 inherit peer-session CORE1`	• The example configures a router to send the peer session template named CORE1 to the 172.16.0.1 neighbor to inherit. This template can be applied to a neighbor, and if another peer session template is indirectly inherited in CORE1, the indirectly inherited configuration will also be applied. No additional peer session templates can be directly applied. However, the directly inherited template can also inherit up to seven additional indirectly inherited peer session templates.
Step 6	**end** **Example:** `Router(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.
Step 7	**show ip bgp template peer-session** [*session-template-name*] **Example:** `Router# show ip bgp template peer-session`	Displays locally configured peer session templates. • The output can be filtered to display a single peer policy template with the optional *session-template-name* argument. This command also supports all standard output modifiers.

What to Do Next

To create a peer policy template, go to the Configuring Peer Policy Templates, on page 125.

Configuring Peer Policy Templates

Configuring Basic Peer Policy Templates

Perform this task to create a basic peer policy template with BGP policy configuration commands that can be applied to many neighbors using one of the next two tasks.

Note The commands in Steps 5 through 7 are optional and could be replaced with any supported BGP policy configuration commands.

Note The following restrictions apply to the peer policy templates:

- A peer policy template can directly or indirectly inherit up to eight peer policy templates.
- A BGP neighbor cannot be configured to work with both peer groups and peer templates. A BGP neighbor can be configured to belong only to a peer group or to inherit policies only from peer templates.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-policy** *policy-template-name*
5. **maximum-prefix** *prefix-limit* [*threshold*] [**restart** *restart-interval* | **warning-only**]
6. **weight** *weight-value*
7. **prefix-list** *prefix-list-name* {**in** | **out**}
8. **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	**template peer-policy** *policy-template-name* **Example:** `Device(config-router)# template peer-policy GLOBAL`	Enters policy-template configuration mode and creates a peer policy template.
Step 5	**maximum-prefix** *prefix-limit* [*threshold*] [**restart** *restart-interval* | **warning-only**] **Example:** `Device(config-router-ptmp)# maximum-prefix 10000`	(Optional) Configures the maximum number of prefixes that a neighbor will accept from this peer. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the "Peer Policy Templates" section.
Step 6	**weight** *weight-value* **Example:** `Device(config-router-ptmp)# weight 300`	(Optional) Sets the default weight for routes that are sent from this neighbor. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the "Peer Policy Templates" section.

	Command or Action	Purpose
Step 7	**prefix-list** *prefix-list-name* {**in** | **out**} **Example:** `Device(config-router-ptmp)# prefix-list NO-MARKETING in`	(Optional) Filters prefixes that are received by the router or sent from the router. • The prefix list in the example filters inbound internal addresses. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the "Peer Policy Templates" section.
Step 8	**end** **Example:** `Device(config-router-ptmp)# end`	Exits policy-template configuration mode and returns to privileged EXEC mode.

What to Do Next

After the peer policy template is created, the configuration of the peer policy template can be inherited or applied by another peer policy template. For details about peer policy inheritance, see the "Configuring Peer Policy Template Inheritance with the inherit peer-policy Command" section or the "Configuring Peer Policy Template Inheritance with the neighbor inherit peer-policy Command" section.

Configuring Peer Policy Template Inheritance with the inherit peer-policy Command

This task configures peer policy template inheritance using the **inherit peer-policy**command. It creates and configure a peer policy template and allows it to inherit a configuration from another peer policy template.

When BGP neighbors use inherited peer templates, it can be difficult to determine which policies are associated with a specific template. In Cisco IOS Release 12.0(25)S, 12.4(11)T, 12.2(33)SRB, 12.2(33)SB, and later releases, the **detail** keyword was added to the **show ip bgp template peer-policy** command to display the detailed configuration of local and inherited policies associated with a specific template.

Note The commands in Steps 5 and 6 are optional and could be replaced with any supported BGP policy configuration commands.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-policy** *policy-template-name*
5. **route-map** *map-name* {**in**| **out**}
6. **inherit peer-policy** *policy-template-name* *sequence-number*
7. **end**
8. **show ip bgp template peer-policy** [*policy-template-name*[**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**template peer-policy** *policy-template-name* **Example:** `Router(config-router)# template peer-policy NETWORK1`	Enter policy-template configuration mode and creates a peer policy template.
Step 5	**route-map** *map-name* {**in**| **out**} **Example:** `Router(config-router-ptmp)# route-map ROUTE in`	(Optional) Applies the specified route map to inbound or outbound routes. **Note** Any supported BGP policy configuration command can be used here. For a list of the supported commands, see the Peer Policy Templates, on page 73.
Step 6	**inherit peer-policy** *policy-template-name* *sequence-number* **Example:** `Router(config-router-ptmp)# inherit peer-policy GLOBAL 10`	Configures the peer policy template to inherit the configuration of another peer policy template. • The *sequence-number* argument sets the order in which the peer policy template is evaluated. Like a route map sequence number, the lowest sequence number is evaluated first. • The example configures this peer policy template to inherit the configuration from GLOBAL. If the template created in these steps is applied to a neighbor, the configuration GLOBAL will also be inherited and applied indirectly. Up to six additional peer policy templates can be indirectly inherited from GLOBAL for a total of eight directly applied and indirectly inherited peer policy templates. • This template in the example will be evaluated first if no other templates are configured with a lower sequence number.

	Command or Action	Purpose
Step 7	**end** **Example:** `Router(config-router-ptmp)# end`	Exits policy-template configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp template peer-policy** [*policy-template-name*[**detail**]] **Example:** `Router# show ip bgp template peer-policy NETWORK1 detail`	Displays locally configured peer policy templates. • The output can be filtered to display a single peer policy template with the *policy-template-name* argument. This command also supports all standard output modifiers. • Use the **detail** keyword to display detailed policy information. **Note** The **detail** keyword is supported only in Cisco IOS Release 12.0(25)S, 12.4(11)T, 12.2(33)SRB, 12.2(33)SB, and later releases.

Examples

The following sample output of the **show ip bgp template peer-policy** command with the **detail** keyword displays details of the policy named NETWORK1. The output in this example shows that the GLOBAL template was inherited. Details of route map and prefix list configurations are also displayed.

```
Router# show ip bgp template peer-policy NETWORK1 detail
Template:NETWORK1, index:2.
Local policies:0x1, Inherited polices:0x80840
This template inherits:
  GLOBAL, index:1, seq_no:10, flags:0x1
Locally configured policies:
  route-map ROUTE in
Inherited policies:
  prefix-list NO-MARKETING in
  weight 300
  maximum-prefix 10000
 Template:NETWORK1 <detail>
Locally configured policies:
  route-map ROUTE in
route-map ROUTE, permit, sequence 10
  Match clauses:
    ip address prefix-lists: DEFAULT
ip prefix-list DEFAULT: 1 entries
   seq 5 permit 10.1.1.0/24
  Set clauses:
  Policy routing matches: 0 packets, 0 bytes
Inherited policies:
  prefix-list NO-MARKETING in
ip prefix-list NO-MARKETING: 1 entries
   seq 5 deny 10.2.2.0/24
```

Configuring Peer Policy Template Inheritance with the neighbor inherit peer-policy Command

This task configures a router to send a peer policy template to a neighbor to inherit using the **neighbor inherit peer-policy** command. Perform the following steps to send a peer policy template configuration to a neighbor to inherit.

When BGP neighbors use multiple levels of peer templates, it can be difficult to determine which policies are applied to the neighbor. In Cisco IOS Release 12.0(25)S, 12.4(11)T, 12.2(33)SRB, 12.2(33)SB, and later releases, the **policy** and **detail** keywords were added to the **show ip bgp neighbors** command to display the inherited policies and policies configured directly on the specified neighbor.

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** [**multicast** | **unicast** | **vrf** *vrf-name*]
6. **neighbor** *ip-address* **inherit peer-policy** *policy-template-name*
7. **end**
8. **show ip bgp neighbors** [*ip-address*[**policy** [**detail**]]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.2 remote-as 40000`	Configures a peering session with the specified neighbor. • The explicit **remote-as** statement is required for the **neighbor inherit** statement in Step 6 to work. If a peering is not configured, the specified neighbor in Step 6 will not accept the session template.

	Command or Action	Purpose
Step 5	**address-family ipv4** [**multicast** | **unicast** | **vrf** *vrf-name*] **Example:** `Router(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure a neighbor to accept address family-specific command configurations.
Step 6	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Router(config-router-af)# neighbor 192.168.1.2 inherit peer-policy GLOBAL`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration. • The example configures a router to send the peer policy template named GLOBAL to the 192.168.1.2 neighbor to inherit. This template can be applied to a neighbor, and if another peer policy template is indirectly inherited from GLOBAL, the indirectly inherited configuration will also be applied. Up to seven additional peer policy templates can be indirectly inherited from GLOBAL.
Step 7	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp neighbors** [*ip-address*[**policy** [**detail**]]] **Example:** `Router# show ip bgp neighbors 192.168.1.2 policy`	Displays locally configured peer policy templates. • The output can be filtered to display a single peer policy template with the *policy-template-name* argument. This command also supports all standard output modifiers. • Use the **policy** keyword to display the policies applied to this neighbor per address family. • Use the **detail** keyword to display detailed policy information. • The **policy** and **detail** keywords are supported only in Cisco IOS Release 12.0(25)S, 12.4(11)T, 12.2(33)SRB, 12.2(33)SB, and later releases. **Note** Only the syntax required for this task is shown. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following sample output shows the policies applied to the neighbor at 192.168.1.2. The output displays both inherited policies and policies configured on the neighbor device. Inherited polices are policies that the neighbor inherits from a peer-group or a peer-policy template.

```
Router# show ip bgp neighbors 192.168.1.2 policy
Neighbor: 192.168.1.2, Address-Family: IPv4 Unicast
Locally configured policies:
 route-map ROUTE in
Inherited polices:
 prefix-list NO-MARKETING in
 route-map ROUTE in
 weight 300
 maximum-prefix 10000
```

Monitoring and Maintaining BGP Dynamic Update Groups

Use this task to clear and display information about the processing of dynamic BGP update groups. The performance of BGP update message generation is improved with the use of BGP update groups. With the configuration of the BGP peer templates and the support of the dynamic BGP update groups, the network operator no longer needs to configure peer groups in BGP and can benefit from improved configuration flexibility and system performance. For information about using BGP peer templates, see the "Configuring Peer Session Templates" and "Configuring Peer Policy Templates" sections.

SUMMARY STEPS

1. **enable**
2. **clear ip bgp update-group** [*index-group* | *ip-address*]
3. **show ip bgp replication** [*index-group* | *ip-address*]
4. **show ip bgp update-group** [*index-group* | *ip-address*] [**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 update-group** [*index-group* | *ip-address*] **Example:** `Device# clear ip bgp update-group 192.168.2.2`	Clears BGP update group membership and recalculate BGP update groups. • In the example provided, the membership of neighbor 192.168.2.2 is cleared from an update group.
Step 3	**show ip bgp replication** [*index-group* | *ip-address*] **Example:** `Device# show ip bgp replication`	Displays update replication statistics for BGP update groups.
Step 4	**show ip bgp update-group** [*index-group* | *ip-address*] [**summary**] **Example:**	Displays information about BGP update groups.

	Command or Action	Purpose
	`Device# show ip bgp update-group`	

Troubleshooting Tips

Use the **debug ip bgp groups** command to display information about the processing of BGP update groups. Information can be displayed for all update groups, an individual update group, or a specific BGP neighbor. The output of this command can be very verbose. This command should not be deployed in a production network unless your are troubleshooting a problem.

Configuration Examples for a Basic BGP Network

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

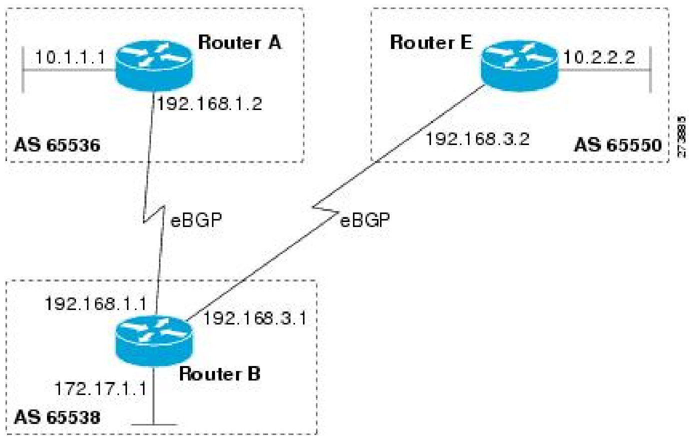
```
router bgp 45000
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 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: Configuring a BGP Routing Process and Peers Using 4-Byte Autonomous System Numbers

Asplain Format

The following example shows the configuration for Router A, Router B, and Router E in the figure below with a Border Gateway Protocol (BGP) process configured between three neighbor peers (at Router A, at Router B, and at Router E) in separate 4-byte autonomous systems configured using asplain notation. IPv4 unicast routes are exchanged with all peers.

Figure 12: BGP Peers Using 4-Byte Autonomous System Numbers in Asplain Format



Router A

```
router bgp 65536
 bgp router-id 10.1.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.1 remote-as 65538
 !
 address-family ipv4
  neighbor 192.168.1.1 activate
  no auto-summary
  no synchronization
  network 10.1.1.0 mask 255.255.255.0
  exit-address-family
```

Router B

```
router bgp 65538
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 65536
 neighbor 192.168.3.2 remote-as 65550
 neighbor 192.168.3.2 description finance
```

```
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

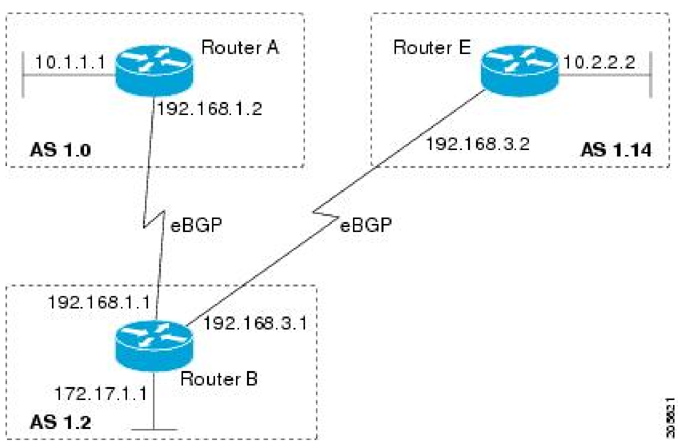
Router E

```
router bgp 65550
 bgp router-id 10.2.2.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.3.1 remote-as 65538
 !
 address-family ipv4
  neighbor 192.168.3.1 activate
  no auto-summary
  no synchronization
  network 10.2.2.0 mask 255.255.255.0
  exit-address-family
```

Asdot Format

The following example shows how to create the configuration for Router A, Router B, and Router E in the figure below with a BGP process configured between three neighbor peers (at Router A, at Router B, and at Router E) in separate 4-byte autonomous systems configured using the default asdot format. IPv4 unicast routes are exchanged with all peers.

Figure 13: BGP Peers Using 4-Byte Autonomous System Numbers in Asdot Format



Router A

```
router bgp 1.0
 bgp router-id 10.1.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
```

```
 timers bgp 70 120
 neighbor 192.168.1.1 remote-as 1.2
 !
 address-family ipv4
  neighbor 192.168.1.1 activate
  no auto-summary
  no synchronization
  network 10.1.1.0 mask 255.255.255.0
  exit-address-family
```

Router B

```
router bgp 1.2
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 1.0
 neighbor 192.168.3.2 remote-as 1.14
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

Router E

```
router bgp 1.14
 bgp router-id 10.2.2.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.3.1 remote-as 1.2
 !
 address-family ipv4
  neighbor 192.168.3.1 activate
  no auto-summary
  no synchronization
  network 10.2.2.0 mask 255.255.255.0
  exit-address-family
```

Examples: Configuring a VRF and Setting an Extended Community Using a BGP 4-Byte Autonomous System Number

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example is available in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases and shows how to create a VRF with a route target that uses a 4-byte autonomous system number, 65537, and how to set the route target to extended community value 65537:100 for routes that are permitted by the route map:

```
ip vrf vpn_red
 rd 64500:100
 route-target both 65537:100
 exit
route-map red_map permit 10
 set extcommunity rt 65537:100
 end
```

After the configuration is completed, use the **show route-map** command to verify that the extended community is set to the route target that contains the 4-byte autonomous system number of 65537:

```
RouterB# show route-map red_map
route-map red_map, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:65537:100
  Policy routing matches: 0 packets, 0 bytes
```

4-Byte Autonomous System Number RD Support

The following example shows how to create a VRF with a route distinguisher that contains a 4-byte AS number 65536, and a route target that contains a 4-byte autonomous system number, 65537:

```
ip vrf vpn_red
 rd 65536:100
 route-target both 65537:100
 exit
```

After the configuration is completed, use the **show vrf** command to verify that the 4-byte AS number route distinguisher is set to 65536:100:

```
RouterB# show vrf vpn_red
Current configuration : 36 bytes
vrf definition x
 rd 65536:100
!
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T

The following example is available in Cisco IOS Release 12.0(32)S12, and 12.4(24)T and shows how to create a VRF with a route target that uses a 4-byte autonomous system number, 1.1, and how to set the route target to the extended community value 1.1:100 for routes that are permitted by the route map.

Note In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, and later releases, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

```
ip vrf vpn_red
 rd 64500:100
 route-target both 1.1:100
 exit
route-map red_map permit 10
 set extcommunity rt 1.1:100
 end
```

After the configuration is completed, use the **show route-map** command to verify that the extended community is set to the route target that contains the 4-byte autonomous system number of 1.1.

```
RouterB# show route-map red_map
route-map red_map, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:1.1:100
  Policy routing matches: 0 packets, 0 bytes
```

Asdot Default Format for 4-Byte Autonomous System Number RD Support

The following example works if you have configured asdot as the default display format using the **bgp asnotation dot** command:

```
ip vrf vpn_red
 rd 1.0:100
 route-target both 1.1:100
 exit
```

Example: NLRI to AFI Configuration

The following example upgrades an existing router configuration file in the NLRI format to the AFI format and set the router CLI to use only commands in the AFI format:

```
router bgp 60000
 bgp upgrade-cli
```

The **show running-config** command can be used in privileged EXEC mode to verify that an existing router configuration file has been upgraded from the NLRI format to the AFI format. The following sections provide sample output from a router configuration file in the NLRI format, and the same router configuration file after it has been upgraded to the AFI format with the **bgp upgrade-cli** command in router configuration mode.

Note After a router has been upgraded from the AFI format to the NLRI format with the **bgp upgrade-cli**command, NLRI commands will no longer be accessible or configurable.

Router Configuration File in NLRI Format Before Upgrading

The following sample output is from the **show running-config** command in privileged EXEC mode. The sample output shows a router configuration file, in the NLRI format, prior to upgrading to the AFI format with the **bgp upgrade-cli** command. The sample output is filtered to show only the affected portion of the router configuration.

```
Router# show running-config | begin bgp

router bgp 101
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.1.1.1 remote-as 505 nlri unicast multicast
 no auto-summary
!
ip default-gateway 10.4.9.1
```

```
ip classless
!
!
route-map REDISTRIBUTE-MULTICAST permit 10
 match ip address prefix-list MULTICAST-PREFIXES
 set nlri multicast
!
route-map MULTICAST-PREFIXES permit 10
!
route-map REDISTRIBUTE-UNICAST permit 20
 match ip address prefix-list UNICAST-PREFIXES
 set nlri unicast
!
!
!
line con 0
line aux 0
line vty 0 4
 password PASSWORD
 login
!
end
```

Router Configuration File in AFI Format After Upgrading

The following sample output shows the router configuration file after it has been upgraded to the AFI format. The sample output is filtered to show only the affected portion of the router configuration file.

```
Router# show running-config | begin bgp

router bgp 101
 bgp log-neighbor-changes
 neighbor 10.1.1.1 remote-as 505
 no auto-summary
 !
 address-family ipv4 multicast
  neighbor 10.1.1.1 activate
  no auto-summary
  no synchronization
  exit-address-family
 !
 address-family ipv4
  neighbor 10.1.1.1 activate
  no auto-summary
  no synchronization
  exit-address-family
!
ip default-gateway 10.4.9.1
ip classless
!
!
route-map REDISTRIBUTE-MULTICAST_mcast permit 10
 match ip address prefix-list MULTICAST-PREFIXES
!
route-map REDISTRIBUTE-MULTICAST permit 10
 match ip address prefix-list MULTICAST-PREFIXES
!
route-map MULTICAST-PREFIXES permit 10
!
route-map REDISTRIBUTE-UNICAST permit 20
 match ip address prefix-list UNICAST-PREFIXES
!
!
```

```
!
line con 0
line aux 0
line vty 0 4
 password PASSWORD
 login
!
end
```

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 BGP Peers Using 4-Byte Autonomous System Numbers

The following examples show how to clear BGP peers belonging to an autonomous system that uses 4-byte autonomous system numbers. The initial state of the BGP routing table is shown using the **show ip bgp** command, and peers in 4-byte autonomous systems 65536 and 65550 are displayed.

```
RouterB# show ip bgp

BGP table version is 4, 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 65536  i
*> 10.2.2.0/24      192.168.3.2              0             0 65550  i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
```

The **clear ip bgp 65550** command is entered to remove all BGP peers in the 4-byte autonomous system 65550. The ADJCHANGE message shows that the BGP peer at 192.168.3.2 is being reset.

```
RouterB# clear ip bgp 65550
RouterB#
*Nov 30 23:25:27.043: %BGP-5-ADJCHANGE: neighbor 192.168.3.2 Down User reset
```

The **show ip bgp** command is entered again, and only the peer in 4-byte autonomous systems 65536 is now displayed.

```
RouterB# show ip bgp

BGP table version is 5, 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 65536  i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
```

Almost immediately, the next ADJCHANGE message shows that the BGP peer at 192.168.3.2 (in the 4-byte autonomous system 65550) is now back up.

```
RouterB#
*Nov 30 23:25:55.995: %BGP-5-ADJCHANGE: neighbor 192.168.3.2 Up
```

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. In Cisco IOS Release 12.2(25)S and later releases, the syntax is **clear ip bgp all**. 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.

```
Router# 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:

```
Router# 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:

```
Router# 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):

```
Router# 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:

```
Router# 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
```

Example: Configuring Peer Session Templates

The following example creates a peer session template named INTERNAL-BGP in session-template configuration mode:

```
router bgp 45000
 template peer-session INTERNAL-BGP
 remote-as 50000
 timers 30 300
 exit-peer-session
```

The following example creates a peer session template named CORE1. This example inherits the configuration of the peer session template named INTERNAL-BGP.

```
router bgp 45000
 template peer-session CORE1
 description CORE-123
 update-source loopback 1
 inherit peer-session INTERNAL-BGP
 exit-peer-session
```

The following example configures the 192.168.3.2 neighbor to inherit the CORE1 peer session template. The 192.168.3.2 neighbor will also indirectly inherit the configuration from the peer session template named INTERNAL-BGP. The explicit **remote-as** statement is required for the neighbor inherit statement to work. If a peering is not configured, the specified neighbor will not accept the session template.

```
router bgp 45000
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 inherit peer-session CORE1
```

Examples: Configuring Peer Policy Templates

The following example creates a peer policy template named GLOBAL and enters policy-template configuration mode:

```
router bgp 45000
 template peer-policy GLOBAL
  weight 1000
  maximum-prefix 5000
  prefix-list NO_SALES in
  exit-peer-policy
```

The following example creates a peer policy template named PRIMARY-IN and enters policy-template configuration mode:

```
router bgp 45000
 template peer-policy PRIMARY-IN
  prefix-list ALLOW-PRIMARY-A in
  route-map SET-LOCAL in
  weight 2345
  default-originate
  exit-peer-policy
```

The following example creates a peer policy template named CUSTOMER-A. This peer policy template is configured to inherit the configuration from the peer policy templates named PRIMARY-IN and GLOBAL.

```
router bgp 45000
 template peer-policy CUSTOMER-A
  route-map SET-COMMUNITY in
  filter-list 20 in
  inherit peer-policy PRIMARY-IN 20
  inherit peer-policy GLOBAL 10
  exit-peer-policy
```

The following example configures the 192.168.2.2 neighbor in address family mode to inherit the peer policy template named CUSTOMER-A. Assuming this example is a continuation of the example above, because the peer policy template named CUSTOMER-A above inherited the configuration from the templates named PRIMARY-IN and GLOBAL, the 192.168.2.2 neighbor will also indirectly inherit the peer policy templates named PRIMARY-IN and GLOBAL.

```
router bgp 45000
 neighbor 192.168.2.2 remote-as 50000
 address-family ipv4 unicast
  neighbor 192.168.2.2 inherit peer-policy CUSTOMER-A
  end
```

Examples: Monitoring and Maintaining BGP Dynamic Update Peer-Groups

No configuration is required to enable the BGP dynamic update of peer groups and the algorithm runs automatically. The following examples show how BGP update group information can be cleared or displayed.

clear ip bgp update-group Example

The following example clears the membership of neighbor 10.0.0.1 from an update group:

```
Router# clear ip bgp update-group 10.0.0.1
```

debug ip bgp groups Example

The following example output from the **debug ip bgp groups** command shows the recalculation of update groups after the **clear ip bgp groups** command was issued:

```
Router# debug ip bgp groups

5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.5 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.5 flags 0x0 cap 0x0 and updgrp 2 fl0
5w4d: BGP-DYN(0): Update-group 2 flags 0x0 cap 0x0 policies same as 10.4.9.5 fl0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.8 flags 0x0 cap 0x0 and updgrp 2 fl0
5w4d: BGP-DYN(0): Update-group 2 flags 0x0 cap 0x0 policies same as 10.4.9.8 fl0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Down User reset
5w4d: BGP-DYN(0): Comparing neighbor 10.4.9.21 flags 0x0 cap 0x0 and updgrp 1 f0
5w4d: BGP-DYN(0): Update-group 1 flags 0x0 cap 0x0 policies same as 10.4.9.21 f0
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.5 Up
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.21 Up
5w4d: %BGP-5-ADJCHANGE: neighbor 10.4.9.8 Up
```

show ip bgp replication Example

The following sample output from the **show ip bgp replication** command shows update group replication information for all for neighbors:

```
Router# show ip bgp replication

BGP Total Messages Formatted/Enqueued : 0/0
     Index      Type  Members          Leader   MsgFmt  MsgRepl  Csize  Qsize
         1 internal         1       10.4.9.21        0        0      0      0
         2 internal         2        10.4.9.5        0        0      0      0
```

show ip bgp update-group Example

The following sample output from the **show ip bgp update-group** command shows update group information for all neighbors:

```
Router# show ip bgp update-group

BGP version 4 update-group 1, internal, Address Family: IPv4 Unicast
  BGP Update version : 0, messages 0/0
  Route map for outgoing advertisements is COST1
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
  Minimum time between advertisement runs is 5 seconds
  Has 1 member:
  10.4.9.21
BGP version 4 update-group 2, internal, Address Family: IPv4 Unicast
  BGP Update version : 0, messages 0/0
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
  Minimum time between advertisement runs is 5 seconds
  Has 2 members:
  10.4.9.5 10.4.9.8
```

Where to Go Next

- If you want to connect to an external service provider, see the "Connecting to a Service Provider Using External BGP" module.
- To configure BGP neighbor session options, proceed to the "Configuring BGP Neighbor Session Options" module.
- If you want to configure some iBGP features, see the "Configuring Internal BGP Features" 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
IPv6 commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS IPv6 Command Reference

Related Topic	Document Title
Overview of Cisco BGP conceptual information with links to all the individual BGP modules	"Cisco BGP Overview" module in the *IP Routing: BGP Configuration Guide*
Multiprotocol Label Switching (MPLS) and BGP configuration example using the IPv4 VRF address family	"MPLS VPN Inter-AS with ASBRs Exchanging IPv4 Routes and MPLS Labels" module in the *MPLS: Layer 3 VPNs: Inter-AS and CSC Configuration Guide*

Standards

Standard	Title
MDT SAFI	MDT SAFI

MIBs

MIB	MIBs Link
CISCO-BGP4-MIB	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 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 of 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)*
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5396	*Textual Representation of Autonomous system (AS) Numbers*
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*

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 Configuring a Basic BGP Network

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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 9: Feature Information for Configuring a Basic BGP Network

Feature Name	Releases	Feature Configuration Information
BGP Conditional Route Injection	12.0(22)S 12.2(4)T 12.2(14)S 15.0(1)S Cisco IOS XE 3.1.0SG	The BGP Conditional Route Injection feature allows you to inject more specific prefixes into a BGP routing table over less specific prefixes that were selected through normal route aggregation. These more specific prefixes can be used to provide a finer granularity of traffic engineering or administrative control than is possible with aggregated routes.
BGP Configuration Using Peer Templates	12.0(24)S 12.2(18)S 12.2(27)SBC 12.3(4)T 15.0(1)S	The BGP Configuration Using Peer Templates feature introduces a new mechanism that groups distinct neighbor configurations for BGP neighbors that share policies. This type of policy configuration has been traditionally configured with BGP peer groups. However, peer groups have certain limitations because peer group configuration is bound to update grouping and specific session characteristics. Configuration templates provide an alternative to peer group configuration and overcome some of the limitations of peer groups.

Feature Name	Releases	Feature Configuration Information
BGP Dynamic Update Peer Groups	12.0(24)S 12.2(18)S 12.2(27)SBC 12.3(4)T 15.0(1)S Cisco IOS XE 3.1.0SG	The BGP Dynamic Update Peer Groups feature introduces a new algorithm that dynamically calculates and optimizes update groups of neighbors that share the same outbound policies and can share the same update messages. In previous versions of Cisco IOS software, BGP update messages were grouped based on peer-group configurations. This method of grouping updates limited outbound policies and specific-session configurations. The BGP Dynamic Update Peer Group feature separates update group replication from peer group configuration, which improves convergence time and flexibility of neighbor configuration.
BGP Hybrid CLI	12.0(22)S 12.2(15)T 15.0(1)S	The BGP Hybrid CLI feature simplifies the migration of BGP networks and existing configurations from the NLRI format to the AFI format. This new functionality allows the network operator to configure commands in the AFI format and save these command configurations to existing NLRI formatted configurations. The feature provides the network operator with the capability to take advantage of new features and provides support for migration from the NLRI format to the AFI format.
Suppress BGP Advertisement for Inactive Routes	12.2(25)S 12.2(33)SXH 15.0(1)M 15.0(1)S	The Suppress BGP Advertisements for Inactive Routes feature allows you to configure the suppression of advertisements for routes that are not installed in the Routing Information Base (RIB). Configuring this feature allows Border Gateway Protocol (BGP) updates to be more consistent with data used for traffic forwarding.

CHAPTER 4

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.
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.

	Command or Action	Purpose
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router)# neighbor 192.168.1.2 route-map LOCAL in`	Applies a route map to incoming or outgoing routes. • In this example, the route map named LOCAL will be applied to incoming routes.
Step 9	Repeat Steps 6 through 8 for every peer that is to be configured with inbound soft reconfiguration.	—
Step 10	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 11	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map LOCAL permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named LOCAL is created.
Step 12	**set ip next-hop** *ip-address* **Example:** `Device(config-route-map)# set ip next-hop 192.168.1.144`	Specifies where output packets that pass a match clause of a route map for policy routing. • In this example, the ip address is set to 192.168.1.144.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp neighbors** [*neighbor-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	(Optional) Displays information about the TCP and BGP connections to neighbors. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 15	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

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

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

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

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

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

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

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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 10: Feature Information for BGP 4 Soft Configuration

Feature Name	Releases	Feature Information
BGP 4 Soft Configuration		BGP 4 Soft Configuration allows BGP4 policies to be configured and activated without clearing the BGP session, hence without invalidating the forwarding cache.

CHAPTER 5

BGP Support for 4-byte ASN

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

Finding Feature Information

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

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

Information About BGP Support for 4-byte ASN

BGP Autonomous System Number Formats

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

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

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

Asdot Only Autonomous System Number Formatting

In Cisco IOS XE Release 2.3, the 4-octet (4-byte) AS numbers are entered and displayed only in asdot notation, for example, 1.10 or 45000.64000. When using regular expressions to match 4-byte AS 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 AS numbers are configured, matched in regular expressions, and displayed in **show** command output in Cisco IOS images where only asdot formatting is available.

Table 11: Asdot Only 4-Byte AS 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 AS Number Formatting

In Cisco IOS XE Release 2.4 and later releases, the Cisco implementation of 4-byte AS numbers uses asplain as the default display format for AS numbers, but you can configure 4-byte AS numbers in both the asplain and asdot format. In addition, the default format for matching 4-byte AS numbers in regular expressions is asplain, so you must ensure that any regular expressions to match 4-byte AS 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 AS 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 AS numbers in either asplain or asdot format, only one format is used to display **show** command output and control 4-byte AS number matching for regular expressions, and the default is asplain format. To display 4-byte AS 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 AS numbers, you can still use 2-byte AS numbers. The **show** command output and regular expression match are not changed and remain in asplain (decimal value) format for 2-byte AS numbers regardless of the format configured for 4-byte AS numbers.

Table 12: Default Asplain 4-Byte AS 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 13: Asdot 4-Byte AS 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 AS Numbers

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

RFC 5398, *Autonomous System (AS) Number Reservation for Documentation Use*, describes new reserved AS 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 AS number registry. Reserved 2-byte AS numbers are in the contiguous block, 64496 to 64511 and reserved 4-byte AS numbers are from 65536 to 65551 inclusive.

Private 2-byte AS numbers are still valid in the range from 64512 to 65534 with 65535 being reserved for special use. Private AS 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 AS numbers to external networks. Cisco IOS software does not remove private AS numbers from routing updates by default. We recommend that ISPs filter private AS numbers.

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

Cisco Implementation of 4-Byte Autonomous System Numbers

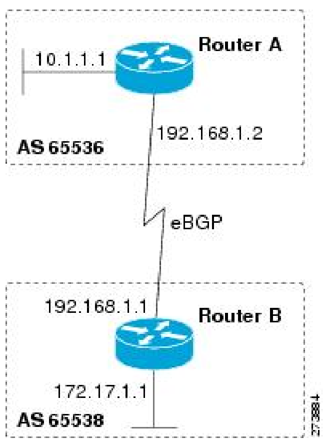
In Cisco IOS XE Release 2.4 and later releases, the Cisco implementation of 4-byte autonomous system (AS) numbers uses asplain—65538, for example—as the default regular expression match and output display format for AS numbers, but you can configure 4-byte AS numbers in both the asplain format and the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

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

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

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

Figure 14: BGP Peers in Two Autonomous Systems Using 4-Byte Numbers



How to Configure BGP Support for 4-byte ASN

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

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

Before you begin

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Device(config-router)# neighbor 192.168.1.2 remote-as 65536`	
Step 5	Repeat Step 4 to define other BGP neighbors, as required.	--
Step 6	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the device is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands.
Step 7	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.2 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local device.
Step 8	Repeat Step 7 to activate other BGP neighbors, as required.	--
Step 9	**network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*] **Example:** `Device(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this AS and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 11	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp 10.1.1.0`	(Optional) Displays the entries in the BGP routing table. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

	Command or Action	Purpose
Step 12	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	(Optional) Displays the status of all BGP connections.

Examples

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

```
RouterB# show ip bgp 10.1.1.0

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

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

```
RouterB# show ip bgp summary

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

Troubleshooting Tips

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays the status of all Border Gateway Protocol (BGP) connections.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 5	**bgp asnotation dot** **Example:** `Device(config-router)# bgp asnotation dot`	Changes the default output format of BGP 4-byte AS numbers from asplain (decimal values) to dot notation. **Note** 4-byte AS numbers can be configured using either asplain format or asdot format. This command affects only the output displayed for **show** commands or the matching of regular expressions.

	Command or Action	Purpose
Step 6	**end** **Example:** `Device(config-router)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 7	**clear ip bgp *** **Example:** `Device# clear ip bgp *`	Clears and resets all current BGP sessions. • In this example, a hard reset is performed to ensure that the 4-byte AS number format change is reflected in all BGP sessions. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 8	**show ip bgp summary** **Example:** `Device# show ip bgp summary`	Displays the status of all BGP connections.
Step 9	**show ip bgp regexp** *regexp* **Example:** `Device# show ip bgp regexp ^1\.0$`	Displays routes that match the AS path regular expression. • In this example, a regular expression to match a 4-byte AS path is configured using asdot format.
Step 10	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 11	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65538`	Enters router configuration mode for the specified routing process. • In this example, the 4-byte AS number, 65538, is defined in asplain notation.
Step 12	**no bgp asnotation dot** **Example:** `Device(config-router)# no bgp asnotation dot`	Resets the default output format of BGP 4-byte AS numbers back to asplain (decimal values). **Note** 4-byte AS numbers can be configured using either asplain format or asdot format. This command affects only the output displayed for **show** commands or the matching of regular expressions.
Step 13	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

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

Examples

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

```
Router# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 65538
BGP table version is 1, main routing table version 1
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  Statd
192.168.1.2     4       65536       7       7        1    0    0 00:03:04      0
192.168.3.2     4       65550       4       4        1    0    0 00:00:15      0
```

After the **bgp asnotation dot** command is configured (followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions), the output is converted to asdot notation format as shown in the following output from the **show ip bgp summary** command. Note the asdot format of the 4-byte AS numbers, 1.0 and 1.14 (these are the asdot conversions of the 65536 and 65550 AS numbers.

```
Router# show ip bgp summary

BGP router identifier 172.17.1.99, local AS number 1.2
BGP table version is 1, main routing table version 1
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  Statd
192.168.1.2     4         1.0       9       9        1    0    0 00:04:13      0
192.168.3.2     4        1.14       6       6        1    0    0 00:01:24      0
```

After the **bgp asnotation dot** command is configured (followed by the **clear ip bgp *** command to perform a hard reset of all current BGP sessions), the regular expression match format for 4-byte AS paths is changed to asdot notation format. Although a 4-byte AS number can be configured in a regular expression using either asplain format or asdot format, only 4-byte AS numbers configured using the current default format are matched. In the first example below, the **show ip bgp regexp** command is configured with a 4-byte AS number in asplain format. The match fails because the default format is currently asdot format and there is no output. In the second example using asdot format, the match passes and the information about the 4-byte AS path is shown using the asdot notation.

Note The asdot notation uses a period, which is a special character in Cisco regular expressions. To remove the special meaning, use a backslash before the period.

```
Router# show ip bgp regexp ^65536$

Router# show ip bgp regexp ^1\.0$

BGP table version is 2, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 1.0 i
```

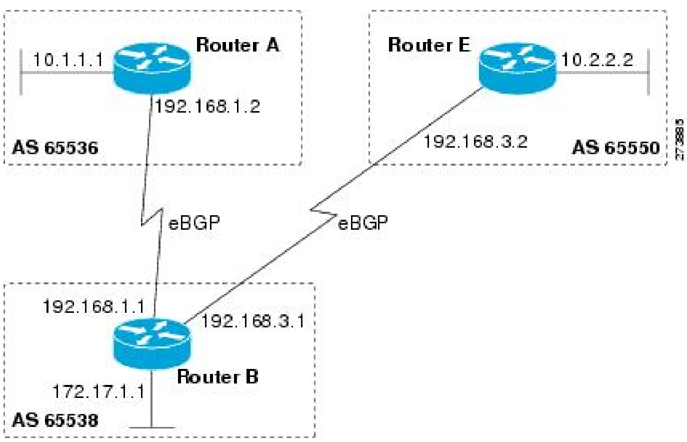
Configuration Examples for BGP Support for 4-byte ASN

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

Asplain Format

The following example shows the configuration for Router A, Router B, and Router E in the figure below with a Border Gateway Protocol (BGP) process configured between three neighbor peers (at Router A, at Router B, and at Router E) in separate 4-byte autonomous systems configured using asplain notation. IPv4 unicast routes are exchanged with all peers.

Figure 15: BGP Peers Using 4-Byte Autonomous System Numbers in Asplain Format



Router A

```
router bgp 65536
 bgp router-id 10.1.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.1 remote-as 65538
 !
 address-family ipv4
```

```
  neighbor 192.168.1.1 activate
  no auto-summary
  no synchronization
  network 10.1.1.0 mask 255.255.255.0
  exit-address-family
```

Router B

```
router bgp 65538
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 65536
 neighbor 192.168.3.2 remote-as 65550
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

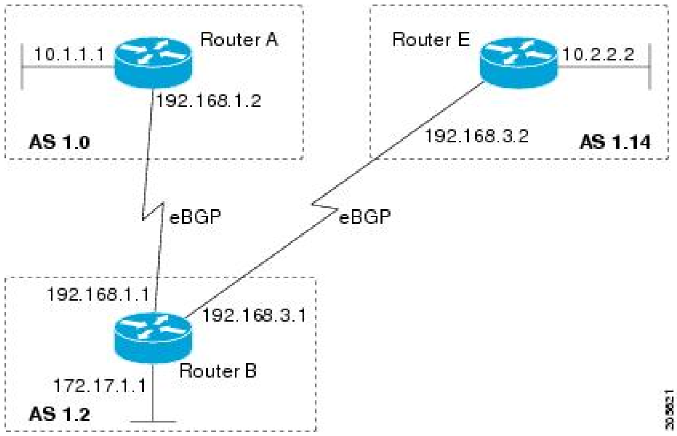
Router E

```
router bgp 65550
 bgp router-id 10.2.2.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.3.1 remote-as 65538
 !
 address-family ipv4
  neighbor 192.168.3.1 activate
  no auto-summary
  no synchronization
  network 10.2.2.0 mask 255.255.255.0
  exit-address-family
```

Asdot Format

The following example shows how to create the configuration for Router A, Router B, and Router E in the figure below with a BGP process configured between three neighbor peers (at Router A, at Router B, and at Router E) in separate 4-byte autonomous systems configured using the default asdot format. IPv4 unicast routes are exchanged with all peers.

Figure 16: BGP Peers Using 4-Byte Autonomous System Numbers in Asdot Format



Router A

```
router bgp 1.0
 bgp router-id 10.1.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.1 remote-as 1.2
 !
 address-family ipv4
  neighbor 192.168.1.1 activate
  no auto-summary
  no synchronization
  network 10.1.1.0 mask 255.255.255.0
  exit-address-family
```

Router B

```
router bgp 1.2
 bgp router-id 172.17.1.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.1.2 remote-as 1.0
 neighbor 192.168.3.2 remote-as 1.14
 neighbor 192.168.3.2 description finance
 !
 address-family ipv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  no auto-summary
  no synchronization
  network 172.17.1.0 mask 255.255.255.0
  exit-address-family
```

Router E

```
router bgp 1.14
```

```
 bgp router-id 10.2.2.99
 no bgp default ipv4-unicast
 bgp fast-external-fallover
 bgp log-neighbor-changes
 timers bgp 70 120
 neighbor 192.168.3.1 remote-as 1.2
 !
 address-family ipv4
  neighbor 192.168.3.1 activate
  no auto-summary
  no synchronization
  network 10.2.2.0 mask 255.255.255.0
  exit-address-family
```

Examples: Configuring a VRF and Setting an Extended Community Using a BGP 4-Byte Autonomous System Number

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example is available in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases and shows how to create a VRF with a route target that uses a 4-byte autonomous system number, 65537, and how to set the route target to extended community value 65537:100 for routes that are permitted by the route map:

```
ip vrf vpn_red
 rd 64500:100
 route-target both 65537:100
 exit
route-map red_map permit 10
 set extcommunity rt 65537:100
 end
```

After the configuration is completed, use the **show route-map** command to verify that the extended community is set to the route target that contains the 4-byte autonomous system number of 65537:

```
RouterB# show route-map red_map
route-map red_map, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:65537:100
  Policy routing matches: 0 packets, 0 bytes
```

4-Byte Autonomous System Number RD Support

The following example shows how to create a VRF with a route distinguisher that contains a 4-byte AS number 65536, and a route target that contains a 4-byte autonomous system number, 65537:

```
ip vrf vpn_red
 rd 65536:100
 route-target both 65537:100
 exit
```

After the configuration is completed, use the **show vrf** command to verify that the 4-byte AS number route distinguisher is set to 65536:100:

```
RouterB# show vrf vpn_red
Current configuration : 36 bytes
vrf definition x
 rd 65536:100
!
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T

The following example is available in Cisco IOS Release 12.0(32)S12, and 12.4(24)T and shows how to create a VRF with a route target that uses a 4-byte autonomous system number, 1.1, and how to set the route target to the extended community value 1.1:100 for routes that are permitted by the route map.

Note In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, and later releases, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

```
ip vrf vpn_red
 rd 64500:100
 route-target both 1.1:100
 exit
route-map red_map permit 10
 set extcommunity rt 1.1:100
 end
```

After the configuration is completed, use the **show route-map** command to verify that the extended community is set to the route target that contains the 4-byte autonomous system number of 1.1.

```
RouterB# show route-map red_map
route-map red_map, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:1.1:100
  Policy routing matches: 0 packets, 0 bytes
```

Asdot Default Format for 4-Byte Autonomous System Number RD Support

The following example works if you have configured asdot as the default display format using the **bgp asnotation dot** command:

```
ip vrf vpn_red
 rd 1.0:100
 route-target both 1.1:100
 exit
```

Additional References for BGP Support for 4-byte ASN

Related Documents

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

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

Standards and RFCs

Standard/RFC	Title
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5396	*Textual Representation of Autonomous System (AS) Numbers*
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*
RFC 5668	*4-Octet AS Specific BGP Extended Community*

Technical Assistance

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

Feature Information for BGP Support for 4-byte ASN

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

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

Table 14: Feature Information for BGP Support for 4-byte ASN

Feature Name	Releases	Feature Information
BGP Support for 4-byte ASN	12.2(33)XNE 12.4(24)T	The BGP Support for 4-Byte ASN feature introduced support for 4-byte autonomous system numbers. The following commands were introduced or modified: **bgp asnotation dot**, **bgp confederation identifier**, **bgp confederation peers**, all **clear ip bgp**commands that configure an autonomous system number, **ip as-path access-list**, **ip extcommunity-list**, **match source-protocol**, **neighbor local-as**, **neighbor remote-as**, **redistribute** (IP), **router bgp**, **route-target**, **set as-path**, **set extcommunity**, **set origin**, all **show ip bgp** commands that display an autonomous system number, and **show ip extcommunity-list**.
BGP—4-Byte ASN RD and RT Support	15.5(1)T	The BGP Support for 4-Byte ASN RD and RT support for 4-byte autonomous system numbers was added.

CHAPTER 6

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.
Step 4	**no bgp default ipv4-unicast** **Example:**	Disables the IPv4 unicast address family for the BGP routing process specified in the previous step.

	Command or Action	Purpose
	`Device(config-router)# no bgp default ipv4-unicast`	**Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** 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.
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.

	Command or Action	Purpose
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:**	Enables privileged EXEC mode. • Enter your password if prompted.

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

	Command or Action	Purpose
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**} **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:cc00::1 route-map rtp in`	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 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.

	Command or Action	Purpose
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

	Command or Action	Purpose
		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 15: Feature Information for IPv6 Routing: Multiprotocol BGP Extensions for IPv6

Feature Name	Releases	Feature Information
IPv6 Routing: Multiprotocol BGP Extensions for IPv6	12.2(2)T 12.3(2)T 12.4(2)T	Multiprotocol BGP Extensions for IPv6 supports the same features and functionality as IPv4 BGP.

CHAPTER 7

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** **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**} **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:**	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.

	Command or Action	Purpose
	`Device(config-router)# neighbor FE80::1234:BFF:FE0E:A471% remote-as 64600`	**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. • 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.

	Command or Action	Purpose
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 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 16: Feature Information for IPv6 Routing: Multiprotocol BGP Link-Local Address Peering

Feature Name	Releases	Feature Information
IPv6 Routing: Multiprotocol BGP Link-Local Address Peering	12.2(4)T 12.3(2)T 12.4(2)T	IPv6 supports multiprotocol BGP link-local address peering.

CHAPTER 8

IPv6 Multicast Address Family Support for Multiprotocol BGP

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 Multicast Address Family Support for Multiprotocol BGP

Multiprotocol BGP for the IPv6 Multicast Address Family

The multiprotocol BGP for the IPv6 multicast address family feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP. IPv6 enhancements to multicast BGP include support for an IPv6 multicast address family and network layer reachability information (NLRI) and next hop (the next router in the path to the destination) attributes that use IPv6 addresses.

Multicast BGP is an enhanced BGP that allows the deployment of interdomain IPv6 multicast. Multiprotocol BGP carries routing information for multiple network layer protocol address families; for example, IPv6 address family and for IPv6 multicast routes. The IPv6 multicast address family contains routes used for RPF lookup by the IPv6 PIM protocol, and multicast BGP IPv6 provides for interdomain transport of the same. Users must use multiprotocol BGP for IPv6 multicast when using IPv6 multicast with BGP because the unicast BGP learned routes will not be used for IPv6 multicast.

Multicast BGP functionality is provided through a separate address family context. A subsequent address family identifier (SAFI) provides information about the type of the network layer reachability information that is carried in the attribute. Multiprotocol BGP unicast uses SAFI 1 messages, and multiprotocol BGP multicast uses SAFI 2 messages. SAFI 1 messages indicate that the routes are usable only for IP unicast, not IP multicast. Because of this functionality, BGP routes in the IPv6 unicast RIB must be ignored in the IPv6 multicast RPF lookup.

A separate BGP routing table is maintained to configure incongruent policies and topologies (for example, IPv6 unicast and multicast) by using IPv6 multicast RPF lookup. Multicast RPF lookup is very similar to the IP unicast route lookup.

No MRIB is associated with the IPv6 multicast BGP table. However, IPv6 multicast BGP operates on the unicast IPv6 RIB when needed. Multicast BGP does not insert or update routes into the IPv6 unicast RIB.

How to Implement IPv6 Multicast Address Family Support for Multiprotocol BGP

Configuring an IPv6 Peer Group to Perform Multicast BGP Routing

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** *as-number*
6. **address-family ipv6** [**unicast** | **multicast**]
7. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **activate**
8. **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *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	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode for the specified BGP routing process.
Step 4	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor group1 peer-group`	Creates a BGP peer group.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *as-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 multicast BGP neighbor table of the local router. • The *ipv6-address* argument in the **neighbor remote-as** command must be in the form documented in RFC 2373 where the address is specified in hexadecimal using 16-bit values between colons.
Step 6	**address-family ipv6** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv6 multicast`	Specifies the IPv6 address family, and enters address family configuration mode. • The **unicast** keyword specifies the IPv6 unicast address family. By default, the router is placed in configuration mode for the IPv6 unicast address family if a keyword is not specified in the **address-family ipv6** command. • The **multicast** keyword specifies IPv6 multicast address prefixes.
Step 7	**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 specified family type with the neighbor and the local router. • To avoid extra configuration steps for each neighbor, use the **neighbor activate** command with the *peer-group-name* argument as an alternative in this step.
Step 8	**neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 2001:DB8:0:CC00::1 peer-group group1`	Assigns the IPv6 address of a BGP neighbor to a peer group.

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

	Command or Action	Purpose
		the address is specified in hexadecimal using 16-bit values between colons. • 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.

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:**	Enters router configuration mode for the specified BGP routing process.

	Command or Action	Purpose
	`Device(config)# router bgp 65000`	
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.

Assigning a BGP Administrative Distance

Caution Changing the administrative distance of BGP internal routes is not recommended. One problem that can occur is the accumulation of routing table inconsistencies, which can break routing.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**}
5. **distance bgp** *external-distance internal-distance local-distance*

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**unicast** | **multicast**} **Example:** `Device(config-router)# address-family ipv6 multicast`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 5	**distance bgp** *external-distance internal-distance local-distance* **Example:** `Device(config-router)# distance bgp 20 20 200`	Assigns a BGP administrative distance.

Generating Translate Updates for IPv6 Multicast BGP

The multicast BGP translate-update feature generally is used in a multicast BGP-capable router that peers with a customer site that has only a BGP-capable router; the customer site has not or cannot upgrade its router to a multicast BGP-capable image. Because the customer site cannot originate multicast BGP advertisements, the router with which it peers will translate the BGP prefixes into multicast BGP prefixes, which are used for multicast-source RPF lookup.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv6** [**unicast** | **multicast**}
5. **neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**]

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv6** [**unicast** | **multicast**} **Example:** `Device(config-router)# address-family ipv6 multicast`	Enters address family configuration mode for configuring routing sessions such as BGP that use standard IPv6 address prefixes.
Step 5	**neighbor** *ipv6-address* **translate-update ipv6 multicast** [**unicast**] **Example:** `Device(config-router)# neighbor 2001:DB8:7000::2 translate-update ipv6 multicast`	Generates multiprotocol IPv6 BGP updates that correspond to unicast IPv6 updates received from a peer.

Resetting IPv6 BGP Sessions

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **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 bgp ipv6** {**unicast** | **multicast**} {***** | *autonomous-system-number* | *ip-address* | *ipv6-address* | **peer-group** *peer-group-name*} [**soft**] [**in** | **out**] **Example:**	Resets IPv6 BGP sessions.

	Command or Action	Purpose
	`Device# clear bgp ipv6 unicast peer-group marketing soft out`	

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.

Clearing IPv6 BGP Route Dampening Information

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** \| **multicast**} **dampening** [*ipv6-prefix*/*prefix-length*]

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**clear bgp ipv6** {**unicast** | **multicast**} **dampening** [*ipv6-prefix*/*prefix-length*] **Example:** `Device# clear bgp ipv6 unicast dampening 2001:DB8::/64`	Clears IPv6 BGP route dampening information and unsuppresses the suppressed routes.

Clearing IPv6 BGP Flap Statistics

SUMMARY STEPS

1. **enable**
2. **clear bgp ipv6** {**unicast** | **multicast**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*]

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**} **flap-statistics** [*ipv6-prefix*/*prefix-length* | **regexp** *regexp* | **filter-list** *list*] **Example:** `Device# clear bgp ipv6 unicast flap-statistics filter-list 3`	Clears IPv6 BGP flap statistics.

Configuration Examples for IPv6 Multicast Address Family Support for Multiprotocol BGP

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: 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: Generating Translate Updates for IPv6 Multicast BGP

The following example shows how to generate IPv6 multicast BGP updates that correspond to unicast IPv6 updates:

```
router bgp 64900
 no bgp default ipv4-unicast
address-family ipv6 multicast
 neighbor 2001:DB8:7000::2 translate-update ipv6 multicast
```

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 Multicast Address Family Support for Multiprotocol BGP

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 IPv6 Multicast Address Family Support for Multiprotocol BGP

Feature Name	Releases	Feature Information
IPv6 Multicast Address Family Support for Multiprotocol BGP	12.0(26)S 12.2(25)S 12.2(25)SG 12.2(33)SRA 12.2(33)SXH 12.3(4)T 12.4 12.4(2)T 15.0(1)S	This feature provides multicast BGP extensions for IPv6 and supports the same features and functionality as IPv4 BGP.

CHAPTER 9

Configuring Multiprotocol BGP (MP-BGP) Support for CLNS

This module describes configuration tasks to configure multiprotocol BGP (MP-BGP) support for CLNS, which provides the ability to scale Connectionless Network Service (CLNS) networks. The multiprotocol extensions of Border Gateway Protocol (BGP) add the ability to interconnect separate Open System Interconnection (OSI) routing domains without merging the routing domains, thus providing the capability to build very large OSI networks.

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.

Restrictions for Configuring MP-BGP Support for CLNS

The configuration of MP-BGP support for CLNS does not support the creation and use of BGP confederations within the CLNS network. We recommend the use of route reflectors to address the issue of a large internal BGP mesh.

BGP extended communities are not supported by this feature.

The following BGP commands are not supported by this feature:

- **auto-summary**
- **neighbor advertise-map**
- **neighbor distribute-list**
- **neighbor soft-reconfiguration**
- **neighbor unsuppress-map**

Information About Configuring MP-BGP Support for CLNS

Address Family Routing Information

By default, commands entered under the **router bgp** command apply to the IPv4 address family. This will continue to be the case unless you enter the **no bgp default ipv4-unicast** command as the first command under the **router bgp** command. The **no bgp default ipv4-unicast** command is configured on the router to disable the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.

Design Features of MP-BGP Support for CLNS

The configuration of MP-BGP support for CLNS allows BGP to be used as an interdomain routing protocol in networks that use CLNS as the network-layer protocol. This feature was developed to solve a scaling issue with a data communications network (DCN) where large numbers of network elements are managed remotely. For details about the DCN issues, see the “DCN Network Topology” section later in this module.

BGP, as an Exterior Gateway Protocol, was designed to handle the volume of routing information generated by the Internet. Network administrators can control the BGP routing information because BGP neighbor relationships (peering) are manually configured and routing updates use incremental broadcasts. Some interior routing protocols such as Intermediate System-to-Intermediate System (IS-IS), in contrast, use a form of automatic neighbor discovery technique and broadcast updates at regular intervals.

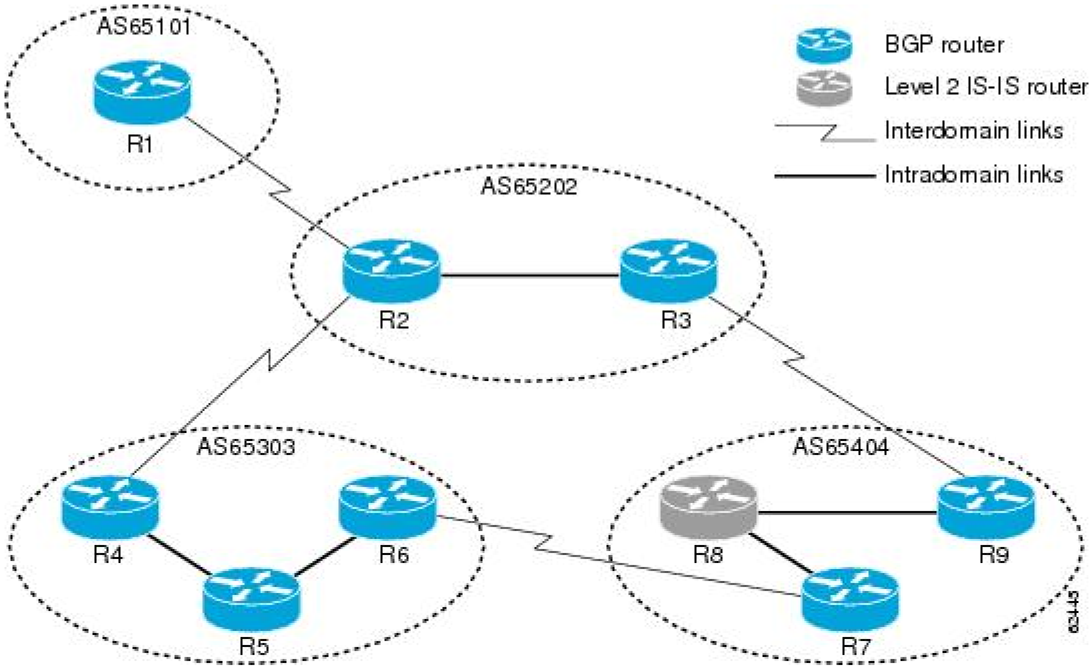
CLNS uses network service access point (NSAP) addresses to identify all its network elements. Using the BGP address-family support, NSAP address prefixes can be transported using BGP. In CLNS, BGP prefixes are inserted into the CLNS Level 2 prefix table. This functionality allows BGP to be used as an interdomain routing protocol between separate CLNS routing domains.

Implementing BGP in routers at the edge of each internal network means that the existing interior protocols need not be changed, minimizing disruption in the network.

Generic BGP CLNS Network Topology

The figure below shows a generic BGP CLNS network containing nine routers that are grouped into four different autonomous systems (in BGP terminology) or routing domains (in OSI terminology). To avoid confusion, we will use the BGP terminology of autonomous systems because each autonomous system is numbered and therefore more easily identified in the diagram and in the configuration discussion.

Figure 17: Components in a Generic BGP CLNS Network



Within each autonomous system, IS-IS is used as the intradomain routing protocol. Between autonomous systems, BGP and its multiprotocol extensions are used as the interdomain routing protocol. Each router is running either a BGP or Level 2 IS-IS routing process. To facilitate this feature, the BGP routers are also running a Level 2 IS-IS process. Although the links are not shown in the figure, each Level 2 IS-IS router is connected to multiple Level 1 IS-IS routers that are, in turn, connected to multiple CLNS networks.

Each autonomous system in this example is configured to demonstrate various BGP features and how these features work with CLNS to provide a scalable interdomain routing solution. In the figure above, the autonomous system AS65101 has a single Level 2 IS-IS router, R1, and is connected to just one other autonomous system, AS65202. Connectivity to the rest of the network is provided by R2, and a default route is generated for R1 to send to R2 all packets with destination NSAP addresses outside of AS65101.

In AS65202 there are two routers, R2 and R3, both with different external BGP (eBGP) neighbors. Routers R2 and R3 are configured to run internal BGP (iBGP) over the internal connection between them.

AS65303 shows how the use of BGP peer groups and route reflection can minimize the need for TCP connections between routers. Fewer connections between routers simplifies the network design and the amount of traffic in the network.

AS65404 shows how to use redistribution to communicate network reachability information to a Level 2 IS-IS router that is not running BGP.

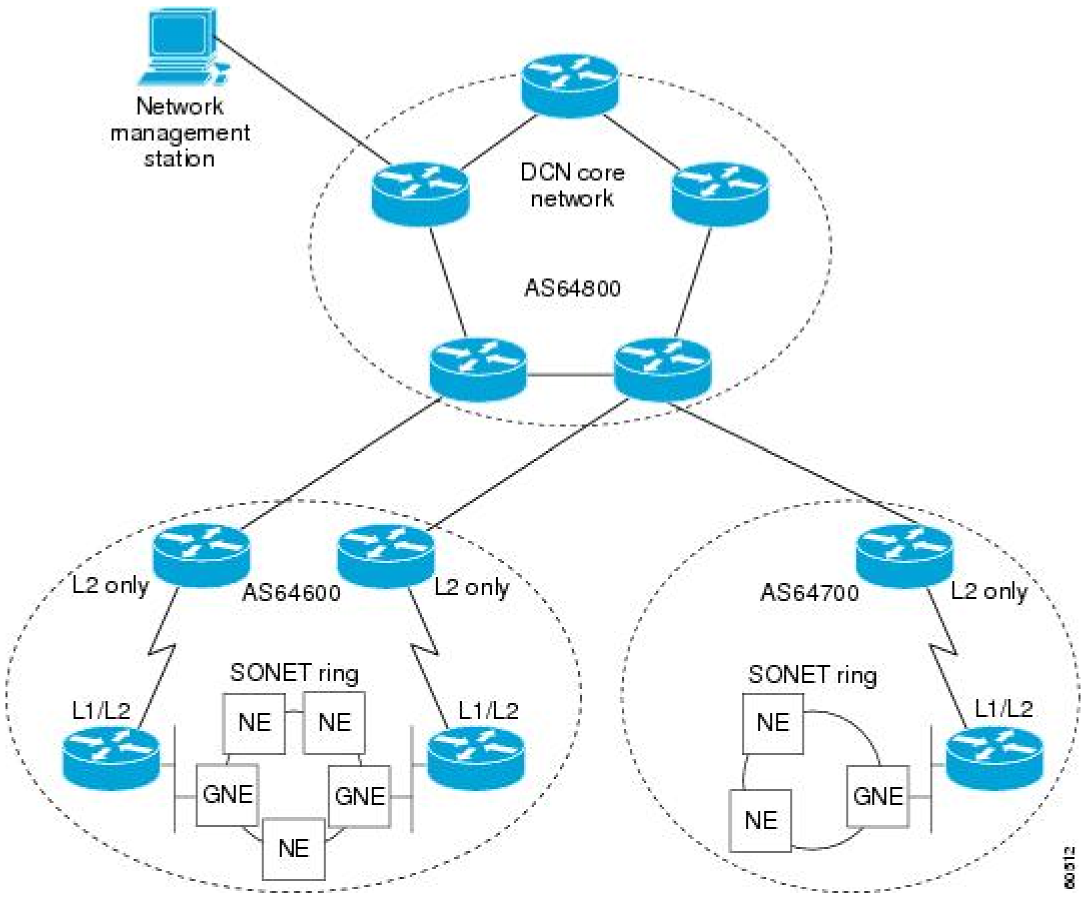
The configuration tasks and examples are based on the generic network design shown in the figure above. Configurations for all the routers in the figure above are listed in .

DCN Network Topology

The Multiprotocol BGP (MP-BGP) Support for CLNS feature can benefit a DCN managing a large number of remote SONET rings. SONET is typically used by telecommunications companies to send data over fiber-optic networks.

The figure below shows some components of a DCN network. To be consistent with the BGP terminology, the figure contains labels to indicate three autonomous systems instead of routing domains. The network elements--designated by NE in Figure 2--of a SONET ring are managed by OSI protocols such as File Transfer, Access, and Management (FTAM) and Common Management Information Protocol (CMIP). FTAM and CMIP run over the CLNS network-layer protocol, which means that the routers providing connectivity must run an OSI routing protocol.

Figure 18: Components in a DCN Network



IS-IS is a link-state protocol used in this example to route CLNS. Each routing node (networking device) is called an intermediate system (IS). The network is divided into areas defined as a collection of routing nodes. Routing within an area is referred to as Level 1 routing. Routing between areas involves Level 2 routing. Routers that link a Level 1 area with a Level 2 area are defined as Level 1-2 routers. A network element that connects to the Level 2 routers that provide a path to the DCN core is represented by a gateway network element--GNE in Figure 2. The network topology here is a point-to-point link between each network element router. In this example, a Level 1 IS-IS router is called an NE router.

Smaller Cisco routers such as the Cisco 2600 series were selected to run as the Level 1-2 routers because shelf space in the central office (CO) of a service provider is very expensive. A Cisco 2600 series router has limited processing power if it is acting as the Level 1 router for four or five different Level 1 areas. The number of Level 1 areas under this configuration is limited to about 200. The entire Level 2 network is also limited by the speed of the slowest Level 2 router.

To provide connectivity between NE routers, in-band signaling is used. The in-band signaling is carried in the SONET/Synchronous Digital Hierarchy (SDH) frame on the data communications channel (DCC). The

DCC is a 192-KB channel, which is a very limited amount of bandwidth for the management traffic. Due to the limited signaling bandwidth between network elements and the limited amount of processing power and memory in the NE routers running IS-IS, each area is restricted to a maximum number of 30 to 40 routers. On average, each SONET ring consists of 10 to 15 network elements.

With a maximum of 200 areas containing 10 to 15 network elements per area, the total number of network element routers in a single autonomous system must be fewer than 3000. Service providers are looking to implement over 10,000 network elements as their networks grow, but the potential number of network elements in an area is limited. The current solution is to break down the DCN into a number of smaller autonomous systems and connect them using static routes or ISO Interior Gateway Routing Protocol (IGRP). ISO IGRP is a proprietary protocol that can limit future equipment implementation options. Static routing does not scale because the growth in the network can exceed the ability of a network administrator to maintain the static routes. BGP has been shown to scale to over 100,000 routes.

To implement the Multiprotocol BGP (MP-BGP) Support for CLNS feature in this example, configure BGP to run on each router in the DCN core network--AS64800 in Figure 2--to exchange routing information between all the autonomous systems. In the autonomous systems AS64600 and AS64700, only the Level 2 routers will run BGP. BGP uses TCP to communicate with BGP-speaking neighbor routers, which means that both an IP-addressed network and an NSAP-addressed network must be configured to cover all the Level 2 IS-IS routers in the autonomous systems AS64600 and AS64700 and all the routers in the DCN core network.

Assuming that each autonomous system--for example, AS64600 and AS64700 in Figure 2--remains the same size with up to 3000 nodes, we can demonstrate how large DCN networks can be supported with this feature. Each autonomous system advertises one address prefix to the core autonomous system. Each address prefix can have two paths associated with it to provide redundancy because there are two links between each autonomous system and the core autonomous system. BGP has been shown to support 100,000 routes, so the core autonomous system can support many other directly linked autonomous systems because each autonomous system generates only a few routes. We can assume that the core autonomous system can support about 2000 directly linked autonomous systems. With the hub-and-spoke design where each autonomous system is directly linked to the core autonomous system, and not acting as a transit autonomous system, the core autonomous system can generate a default route to each linked autonomous system. Using the default routes, the Level 2 routers in the linked autonomous systems process only a small amount of additional routing information. Multiplying the 2000 linked autonomous systems by the 3000 nodes within each autonomous system could allow up to 6 million network elements.

Benefits of MP-BGP Support for CLNS

The Multiprotocol BGP (MP-BGP) Support for CLNS feature adds the ability to interconnect separate OSI routing domains without merging the routing domains, which provides the capability to build very large OSI networks. The benefits of using this feature are not confined to DCN networks, and can be implemented to help scale any network using OSI routing protocols with CLNS.

How to Configure MP-BGP Support for CLNS

Configuring and Activating a BGP Neighbor to Support CLNS

To configure and activate a BGP routing process and an associated BGP neighbor (peer) to support CLNS, perform the steps in this procedure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number*
6. **address-family nsap** [**unicast**]
7. **neighbor** *ip-address* **activate**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65101`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument identifies the autonomous system in which the router resides. Valid values are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config-router)# neighbor 10.1.2.2 remote-as 64202`	Adds an IP address or peer group name of the BGP neighbor in the specified autonomous system to the BGP neighbor table of the local router.
Step 6	**address-family nsap** [**unicast**] **Example:** `Router(config-router)#`	Specifies the NSAP address family and enters address family configuration mode. • The optional **unicast** keyword specifies the NSAP unicast address prefixes. By default, the router is

	Command or Action	Purpose
	`address-family nsap`	placed in configuration mode for the unicast NSAP address family if the **unicast** keyword is not specified with the **address-family nsap** command.

Step 7	**neighbor** *ip-address* **activate**
	Example:
	`Router(config-router-af)#`
	`neighbor 10.1.2.2 activate`
		Enables the BGP neighbor to exchange prefixes for the NSAP address family with the local router.
		Note If you have configured a peer group as a BGP neighbor, you do not use this command because peer groups are automatically activated when any peer group parameter is configured.

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

Configuring an IS-IS Routing Process

When an integrated IS-IS routing process is configured, the first instance of the IS-IS routing process configured is by default a Level 1-2 (intra-area and interarea) router. All subsequent IS-IS routing processes on a network running CLNS are configured as Level 1. All subsequent IS-IS routing processes on a network running IP are configured as Level-1-2. To use the Multiprotocol BGP (MP-BGP) Support for CLNS feature, configure a Level 2 routing process.

To configure an IS-IS routing process and assign it as a Level-2-only process, perform the steps in this procedure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **net** *network-entity-title*
5. **is-type** [**level-1** | **level-1-2** | **level-2-only**]
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis osi-as-101`	Configures an IS-IS routing process and enters router configuration mode for the specified routing process. • The *area-tag* argument is a meaningful name for a routing process. It must be unique among all IP and CLNS routing processes for a given router.
Step 4	**net** *network-entity-title* **Example:** `Router(config-router)# net 49.0101.1111.1111.1111.1111.00`	Configures a network entity title (NET) for the routing process. If you are configuring multiarea IS-IS, you must specify a NET for each routing process.
Step 5	**is-type** [**level-1** | **level-1-2** | **level-2-only**] **Example:** `Router(config-router)# is-type level-1`	Configures the router to act as a Level 1 (intra-area) router, as both a Level 1 router and a Level 2 (interarea) router, or as an interarea router only. • In multiarea IS-IS configurations, the first instance of the IS-IS routing process configured is by default a Level 1-2 (intra-area and interarea) router. All subsequent IS-IS routing processes on a network running CLNS are configured as Level 1. All subsequent IS-IS routing processes on a network running IP are configured as Level-1-2.

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

Configuring Interfaces That Connect to BGP Neighbors

When a router running IS-IS is directly connected to an eBGP neighbor, the interface between the two eBGP neighbors is activated using the **clns enable** command, which allows CLNS packets to be forwarded across the interface. The **clns enable** command activates the End System-to-Intermediate System (ES-IS) protocol to search for neighboring OSI systems.

Note Running IS-IS across the same interface that is connected to an eBGP neighbor can lead to undesirable results if the two OSI routing domains merge into a single domain.

When a neighboring OSI system is found, BGP checks that it is also an eBGP neighbor configured for the NSAP address family. If both the preceding conditions are met, BGP creates a special BGP neighbor route in the CLNS Level 2 prefix routing table. The special BGP neighbor route is automatically redistributed in to the Level 2 routing updates so that all other Level 2 IS-IS routers in the local OSI routing domain know how to reach this eBGP neighbor.

To configure interfaces that are being used to connect with eBGP neighbors, perform the steps in this procedure. These interfaces will normally be directly connected to their eBGP neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **clns enable**
6. **no shutdown**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface serial 2/0`	Specifies the interface type and number and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 10.1.2.2 255.255.255.0`	Configures the interface with an IP address.

Step 5	**clns enable**
	Example:
	`Router(config-if)#`
	`clns enable`
		Specifies that CLNS packets can be forwarded across this interface. The ES-IS protocol is activated and starts to search for adjacent OSI systems.

	Command or Action	Purpose
Step 6	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Turns on the interface.
Step 7	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Interfaces Connected to the Local OSI Routing Domain

To configure interfaces that are connected to the local OSI routing domain, perform the steps in this procedure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **clns router isis** *area-tag*
6. **ip router isis** *area-tag*
7. **no shutdown**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 0/1`	Specifies the interface type and number and enters interface configuration mode.
Step 4	**ip address** *ip-address mask* **Example:**	Configures the interface with an IP address.

	Command or Action	Purpose
	`Router(config-if)# ip address 10.2.3.1 255.255.255.0`	**Note** This step is required only when the interface needs to communicate with an iBGP neighbor.

Step 5	**clns router isis** *area-tag*
	Example:
	`Router(config-if)#`
	`clns router isis osi-as-202`
		Specifies that the interface is actively routing IS-IS when the network protocol is ISO CLNS and identifies the area associated with this routing process.

Step 6	**ip router isis** *area-tag* **Example:** `Router(config-if)# ip router isis osi-as-202`	Specifies that the interface is actively routing IS-IS when the network protocol is IP and identifies the area associated with this routing process. **Note** This step is required only when the interface needs to communicate with an iBGP neighbor, and the IGP is IS-IS.
Step 7	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Turns on the interface.
Step 8	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Advertising Networking Prefixes

Advertising NSAP address prefix forces the prefixes to be added to the BGP routing table. To configure advertisement of networking prefixes, perform the steps in this procedure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number*
6. **address-family nsap** [**unicast**]
7. **network** *nsap-prefix* [**route-map** *map-tag*]
8. **neighbor** *ip-address* **activate**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65101`	Configures a BGP routing process and enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config-router)# neighbor 10.1.2.2 remote-as 64202`	Adds an IP address or peer group name of the BGP neighbor in the specified autonomous system to the BGP neighbor table of the local router.
Step 6	**address-family nsap** [**unicast**] **Example:** `Router(config-router)# address-family nsap`	Specifies the NSAP address family and enters address family configuration mode. • The optional **unicast** keyword specifies the NSAP unicast address prefixes. By default, the router is placed in unicast NSAP address family configuration mode if the **unicast** keyword is not specified with the **address-family nsap** command.
Step 7	**network** *nsap-prefix* [**route-map** *map-tag*] **Example:** `Router(config-router-af)# network 49.0101.1111.1111.1111.1111.00`	Advertises a single prefix of the local OSI routing domain and enters it in the BGP routing table. **Note** It is possible to advertise a single prefix, in which case this prefix could be the unique NSAP address prefix of the local OSI routing domain. Alternatively, multiple longer prefixes, each covering a small portion of the OSI routing domain, can be used to selectively advertise different areas.

	Command or Action	Purpose
		• The advertising of NSAP address prefixes can be controlled by using the optional **route-map** keyword. If no route map is specified, all NSAP address prefixes are redistributed.
Step 8	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af) neighbor 10.1.2.2 activate`	Specifies that NSAP routing information will be sent to the specified BGP neighbor. **Note** See the description of the **neighbor** command in the documents listed in the "Additional References" for more details on the use of this command.

Step 9	**end**
	Example:
	`Router(config-router-af)#`
	`end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Redistributing Routes from BGP into IS-IS

Route redistribution must be approached with caution. We do not recommend injecting the full set of BGP routes into IS-IS because excessive routing traffic will be added to IS-IS. Route maps can be used to control which dynamic routes are redistributed.

To configure route redistribution from BGP into IS-IS, perform the steps in this procedure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router isis** *area-tag*
4. **net** *network-entity-title*
5. **redistribute** *protocol as-number* [*route-type*] [**route-map** *map-tag*]
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**router isis** *area-tag* **Example:** `Router(config)# router isis osi-as-404`	Configures an IS-IS routing process and enters router configuration mode for the specified routing process. **Note** You cannot redistribute BGP routes into a Level 1-only IS-IS routing process.
Step 4	**net** *network-entity-title* **Example:** `Router(config-router)# net 49.0404.7777.7777.7777.7777.00`	Configures a network entity title (NET) for the routing process. If you are configuring multiarea IS-IS, you must specify a NET for each routing process.
Step 5	**redistribute** *protocol* *as-number* [*route-type*] [**route-map** *map-tag*] **Example:** `Router(config-router)#  redistribute bgp 65404 clns`	Redistributes NSAP prefix routes from BGP into the CLNS Level 2 routing table associated with the IS-IS routing process when the *protocol* argument is set to **bgp**and the *route-type* argument is set to **clns**. • The *as-number* argument is defined as the autonomous system number of the BGP routing process to be redistributed into CLNS. • The redistribution of routes can be controlled by using the optional **route-map** keyword. If no route map is specified, all BGP routes are redistributed.

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

Redistributing Routes from IS-IS into BGP

Route redistribution must be approached with caution because redistributed route information is stored in the routing tables. Large routing tables may make the routing process slower. Route maps can be used to control which dynamic routes are redistributed.

To configure route redistribution from IS-IS into BGP, perform the steps in this procedure.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **address-family nsap** [**unicast**]

6. **redistribute** *protocol* [*process-id*] [*route-type*] [**route-map** *map-tag*]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65202`	Configures a BGP routing process and enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.

Step 5	**address-family nsap** [**unicast**]
	Example:
	`Router(config-router)#`
	`address-family nsap`
		Specifies the NSAP address family and enters address family configuration mode.

Step 6	**redistribute** *protocol* [*process-id*] [*route-type*] [**route-map** *map-tag*] **Example:** `Router(config-router-af)# redistribute isis osi-as-202 clns route-map internal-routes-only`	Redistributes routes from the CLNS Level 2 routing table associated with the IS-IS routing process into BGP as NSAP prefixes when the *protocol* argument is set to **isis**and the *route-type* argument is set to **clns**. • The *process-id* argument is defined as the area name for the relevant IS-IS routing process to be redistributed. • The redistribution of routes can be controlled by using the optional **route-map** keyword. If no route map is specified, all Level 2 routes are redistributed.
Step 7	**end** **Example:** `Router(config-router-af)#`	Exits address family configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`end`	

Configuring BGP Peer Groups and Route Reflectors

BGP peer groups reduce the number of configuration commands by applying a BGP **neighbor** command to multiple neighbors. Using a BGP peer group with a local router configured as a BGP route reflector allows BGP routing information received from one member of the group to be replicated to all other group members. Without a peer group, each route reflector client must be specified by IP address.

To create a BGP peer group and use the group as a BGP route reflector client, perform the steps in this procedure. This is an optional task and is used with internal BGP neighbors. In this task, some of the BGP syntax is shown with the *peer-group-name* argument only and only one neighbor is configured as a member of the peer group. Repeat Step 9 to configure other BGP neighbors as members of the peer group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** *peer-group-name* **peer-group**
6. **neighbor** *peer-group-name* **remote-as** *as-number*
7. **address-family nsap** [**unicast**]
8. **neighbor** *peer-group-name* **route-reflector-client**
9. **neighbor** *ip-address* **peer-group** *peer-group*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65303`	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:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.
Step 5	**neighbor** *peer-group-name* **peer-group** **Example:** `Router(config-router)# neighbor ibgp-peers peer-group`	Creates a BGP peer group.
Step 6	**neighbor** *peer-group-name* **remote-as** *as-number* **Example:** `Router(config-router)# neighbor ibgp-peers remote-as 65303`	Adds the peer group name of the BGP neighbor in the specified autonomous system to the BGP neighbor table of the local router.

Step 7	**address-family nsap** [**unicast**]
	Example:
	`Router(config-router)#`
	`address-family nsap`
		Specifies the NSAP address family and enters address family configuration mode.

Step 8	**neighbor** *peer-group-name* **route-reflector-client**
	Example:
	`Router(config-router-af)#`
	`neighbor ibgp-peers route-reflector-client`
		Configures the router as a BGP route reflector and configures the specified peer group as its client.

Step 9	**neighbor** *ip-address* **peer-group** *peer-group*
	Example:
	`Router(config-router-af)#`
	`neighbor 10.4.5.4 peer-group ibgp-peers`
		Assigns a BGP neighbor to a BGP peer group.

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

Filtering Inbound Routes Based on NSAP Prefixes

Perform this task to filter inbound BGP routes based on NSAP prefixes. The **neighbor prefix-list in** command is configured in address family configuration mode to filter inbound routes.

Before you begin

You must specify either a CLNS filter set or a CLNS filter expression before configuring the **neighbor** command. See descriptions for the **clns filter-expr** and **clns filter-set** commands for more information.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **address-family nsap** [**unicast**]
6. **neighbor** {*ip-address*| *peer-group-name*}**prefix-list** {*clns-filter-expr-name*| *clns-filter-set-name*} **in**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65200`	Configures a BGP routing process and enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.

Step 5	**address-family nsap** [**unicast**]
	Example:
	`Router(config-router)#`
	`address-family nsap`
		Specifies the address family and enters address family configuration mode.

	Command or Action	Purpose
Step 6	**neighbor** {*ip-address*| *peer-group-name*}**prefix-list** {*clns-filter-expr-name*| *clns-filter-set-name*} **in** **Example:** `Router(config-router-af)# neighbor 10.23.4.1 prefix-list abc in`	Specifies a CLNS filter set or CLNS filter expression to be used to filter inbound BGP routes. • The *clns-filter-expr-name* argument is defined with the **clns filter-expr** configuration command. • The *clns-filter-set-name* argument is defined with the **clns filter-set** configuration command.

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

Filtering Outbound BGP Updates Based on NSAP Prefixes

Perform this task to filter outbound BGP updates based on NSAP prefixes, use the **neighbor prefix-list out** command in address family configuration mode. This task is configured at Router 7 in the figure above (in the "Generic BGP CLNS Network Topology" section). In this task, a CLNS filter is created with two entries to deny NSAP prefixes starting with 49.0404 and to permit all other NSAP prefixes starting with 49. A BGP peer group is created and the filter is applied to outbound BGP updates for the neighbor that is a member of the peer group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **clns filter-set** *name* [**deny**] **template**
4. **clns filter-set** *name* [**permit**] **template**
5. **router bgp** *as-number*
6. **no bgp default ipv4-unicast**
7. **neighbor** *peer-group-name* **peer-group**
8. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number*
9. **address-family nsap** [**unicast**]
10. **neighbor** {*ip-address* | *peer-group-name*} **prefix-list** {*clns-filter-expr-name* | *clns-filter-set-name*} **out**
11. **neighbor** *ip-address* **peer-group** *peer-group*
12. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**clns filter-set** *name* [**deny**] **template**
	Example:
	`Router(config)#`
	`clns filter-set routes0404 deny 49.0404...`
		Defines a NSAP prefix match for a deny condition for use in CLNS filter expressions.
		• In this example, a deny action is returned if an address starts with 49.0404.

Step 4	**clns filter-set** *name* [**permit**] **template**
	Example:
	`Router(config)#`
	`clns filter-set routes0404 permit 49...`
		Defines a NSAP prefix match for a permit condition for use in CLNS filter expressions.
		• In this example, a permit action is returned if an address starts with 49.
		Note Although the permit example in this step allows all NSAP addresses starting with 49, the match condition in Step 3 is processed first so the NSAP addresses starting with 49.0404 are still denied.

Step 5	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65404`	Configures a BGP routing process and enters router configuration mode for the specified routing process.
Step 6	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.
Step 7	**neighbor** *peer-group-name* **peer-group** **Example:** `Router(config-router)# neighbor ebgp-peers peer-group`	Creates a BGP peer group. • In this example, the BGP peer group named ebgp-peers is created.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Router(config-router)# neighbor ebgp-peers remote-as 65303`	Adds an IP address or peer group name of the BGP neighbor in the specified autonomous system to the BGP neighbor table of the local router. • In this example, the peer group named ebgp-peers is added to the BGP neighbor table.

	Command or Action	Purpose

Step 9	**address-family nsap** [**unicast**]
	Example:
	`Router(config-router)#`
	`address-family nsap`
		Specifies the NSAP address family and enters address family configuration mode.

Step 10	**neighbor** {*ip-address* | *peer-group-name*} **prefix-list** {*clns-filter-expr-name* | *clns-filter-set-name*} **out**
	Example:
	`Router(config-router-af)#`
	`neighbor ebgp-peers prefix-list routes0404 out`
		Specifies a CLNS filter set or CLNS filter expression to be used to filter outbound BGP updates.
		• The *clns-filter-expr-name* argument is defined with the **clns filter-expr** configuration command.
		• The *clns-filter-set-name* argument is defined with the **clns filter-set** configuration command.
		• In this example, the filter set named routes0404 was created in Step3 and Step 4.

Step 11	**neighbor** *ip-address* **peer-group** *peer-group*
	Example:
	`Router(config-router-af)#`
	`neighbor 10.6.7.8 peer-group ebgp-peers`
		Assigns a BGP neighbor to a BGP peer group.

Step 12	**end**
	Example:
	`Router(config-router-af)#`
	`end`
		Exits address family configuration mode and returns to privileged EXEC mode.

Originating Default Routes for a Neighboring Routing Domain

To create a default CLNS route that points to the local router on behalf of a neighboring OSI routing domain, perform the steps in this procedure. This is an optional task and is normally used only with external BGP neighbors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **address-family nsap** [**unicast**]
6. **neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** map-tag]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 64803`	Configures a BGP routing process and enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.

Step 5	**address-family nsap** [**unicast**]
	Example:
	`Router(config-router)#`
	`address-family nsap`
		Specifies the NSAP address family and enters address family configuration mode.

Step 6	**neighbor** {*ip-address* | *peer-group-name*} **default-originate** [**route-map** map-tag]
	Example:
	`Router(config-router-af)#`
	` neighbor 172.16.2.3 default-originate`
		Generates a default CLNS route that points to the local router and that will be advertised to the neighboring OSI routing domain.

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

Verifying MP-BGP Support for CLNS

To verify the configuration, use the **show running-config** EXEC command. Sample output is located in the Example: Implementing MP-BGP Support for CLNS , on page 241. To verify that the Multiprotocol BGP (MP-BGP) Support for CLNS feature is working, perform the following steps.

SUMMARY STEPS

1. **show clns neighbors**
2. **show clns route**
3. **show bgp nsap unicast summary**
4. Enter the **show bgp nsap unicast** command to display all the NSAP prefix routes that the local router has discovered. In the following example of output from router R2, shown in the figure above (in the "Generic BGP CLNS Network Topology" section), a single valid route to prefix 49.0101 is shown. Two valid routes--marked by a *--are shown for the prefix 49.0404. The second route is marked with a *>i sequence, representing the best route to this prefix.

DETAILED STEPS

Step 1 **show clns neighbors**

Use this command to confirm that the local router has formed all the necessary IS-IS adjacencies with other Level 2 IS-IS routers in the local OSI routing domain. If the local router has any directly connected external BGP peers, the output from this command will show that the external neighbors have been discovered, in the form of ES-IS adjacencies.

In the following example, the output is displayed for router R2, shown in the figure above (in the "Generic BGP CLNS Network Topology" section). R2 has three CLNS neighbors. R1 and R4 are ES-IS neighbors because these nodes are in different autonomous systems from R2. R3 is an IS-IS neighbor because it is in the same autonomous system as R2. Note that the system ID is replaced by CLNS hostnames (r1, r3, and r4) that are defined at the start of each configuration file. Specifying the CLNS hostname means that you need not remember which system ID corresponds to which hostname.

Example:

```
Router# show clns neighbors
Tag osi-as-202:
System Id      Interface   SNPA                State  Holdtime  Type Protocol
r1             Se2/0       *HDLC*              Up     274       IS   ES-IS
r3             Et0/1       0002.16de.8481      Up     9         L2   IS-IS
r4             Se2/2       *HDLC*              Up     275       IS   ES-IS
```

Step 2 **show clns route**

Use this command to confirm that the local router has calculated routes to other areas in the local OSI routing domain. In the following example of output from router R2, shown in the figure above (in the "Generic BGP CLNS Network Topology" section), the routing table entry--i 49.0202.3333 [110/10] via R3--shows that router R2 knows about other local IS-IS areas within the local OSI routing domain.

Example:

```
Router# show clns route
Codes: C - connected, S - static, d - DecnetIV
       I - ISO-IGRP,  i - IS-IS,  e - ES-IS
       B - BGP,       b - eBGP-neighbor
C  49.0202.2222 [2/0], Local IS-IS Area
C  49.0202.2222.2222.2222.2222.00 [1/0], Local IS-IS NET
b  49.0101.1111.1111.1111.1111.00 [15/10]
     via r1, Serial2/0
i  49.0202.3333 [110/10]
     via r3, Ethernet0/1
b  49.0303.4444.4444.4444.4444.00 [15/10]
     via r4, Serial2/2
B  49.0101 [20/1]
```

```
      via r1, Serial2/0
B  49.0303 [20/1]
      via r4, Serial2/2
B  49.0404 [200/1]
      via r9
i  49.0404.9999.9999.9999.9999.00 [110/10]
      via r3, Ethernet0/1
```

Step 3 **show bgp nsap unicast summary**

Use this command to verify that the TCP connection to a particular neighbor is active. In the following example output, search the appropriate row based on the IP address of the neighbor. If the State/PfxRcd column entry is a number, including zero, the TCP connection for that neighbor is active.

Example:

```
Router# show bgp nsap unicast summary
BGP router identifier 10.1.57.11, local AS number 65202
BGP table version is 6, main routing table version 6
5 network entries and 8 paths using 1141 bytes of memory
6 BGP path attribute entries using 360 bytes of memory
4 BGP AS-PATH entries using 96 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 activity 5/0 prefixes, 8/0 paths, scan interval 60 secs
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.1.2.1        4 65101      34      34        6    0    0 00:29:11        1
10.2.3.3        4 65202      35      36        6    0    0 00:29:16        3
```

Step 4 Enter the **show bgp nsap unicast** command to display all the NSAP prefix routes that the local router has discovered. In the following example of output from router R2, shown in the figure above (in the "Generic BGP CLNS Network Topology" section), a single valid route to prefix 49.0101 is shown. Two valid routes--marked by a *--are shown for the prefix 49.0404. The second route is marked with a *>i sequence, representing the best route to this prefix.

Example:

```
Router# show bgp nsap unicast
BGP table version is 3, local router ID is 192.168.3.1
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
*> 49.0101          49.0101.1111.1111.1111.1111.00
                                                          0 65101 i
* i49.0202.2222     49.0202.3333.3333.3333.3333.00
                                                 100      0 ?
*>                  49.0202.2222.2222.2222.2222.00
                                                      32768 ?
* i49.0202.3333     49.0202.3333.3333.3333.3333.00
                                                 100      0 ?
*>                  49.0202.2222.2222.2222.2222.00
                                                      32768 ?
*> 49.0303          49.0303.4444.4444.4444.4444.00
                                                          0 65303 i
*  49.0404          49.0303.4444.4444.4444.4444.00
                                                          0 65303 65404 i
*>i                 49.0404.9999.9999.9999.9999.00
                                                 100      0 65404 i
```

Troubleshooting MP-BGP Support for CLNS

The **debug bgp nsap unicast**commands enable diagnostic output concerning various events relating to the operation of the CLNS packets in the BGP routing protocol to be displayed on a console. These commands are intended only for troubleshooting purposes because the volume of output generated by the software when they are used can result in severe performance degradation on the router. See the *Cisco IOS Debug Command Reference* for more information about using these **debug** commands.

To troubleshoot problems with the configuration of MP-BGP support for CLNS and to minimize the impact of the **debug**commands used in this procedure, perform the following steps.

SUMMARY STEPS

1. Attach a console directly to a router running the Cisco software release that includes the Multiprotocol BGP (MP-BGP) Support for CLNS feature.
2. **no logging console**
3. Use Telnet to access a router port.
4. **enable**
5. **terminal monitor**
6. **debug bgp nsap unicast** [*neighbor-address* | **dampening** | **keepalives** | **updates**]
7. **no terminal monitor**
8. **no debug bgp nsap unicast** [*neighbor-address* | **dampening** | **keepalives** | **updates**]
9. **logging console**

DETAILED STEPS

Step 1 Attach a console directly to a router running the Cisco software release that includes the Multiprotocol BGP (MP-BGP) Support for CLNS feature.

Note This procedure will minimize the load on the router created by the **debug bgp nsap unicast** commands because the console port will no longer be generating character-by-character processor interrupts. If you cannot connect to a console directly, you can run this procedure via a terminal server. If you must break the Telnet connection, however, you may not be able to reconnect because the router may be unable to respond due to the processor load of generating the **debug bgp nsap unicast** output.

Step 2 **no logging console**

This command disables all logging to the console terminal.

Step 3 Use Telnet to access a router port.

Step 4 **enable**

Enter this command to access privileged EXEC mode.

Step 5 **terminal monitor**

This command enables logging on the virtual terminal.

Step 6 **debug bgp nsap unicast** [*neighbor-address* | **dampening** | **keepalives** | **updates**]

Enter only specific **debug bgp nsap unicast**commands to isolate the output to a certain subcomponent and minimize the load on the processor. Use appropriate arguments and keywords to generate more detailed debug information on specified subcomponents.

Step 7 **no terminal monitor**

This command disables logging on the virtual terminal.

Step 8 **no debug bgp nsap unicast** [*neighbor-address* | **dampening** | **keepalives** | **updates**]

Enter the specific **no debug bgp nsap unicast**command when you are finished.

Step 9 **logging console**

This command reenables logging to the console.

Configuration Examples for MP-BGP Support for CLNS

Example: Configuring and Activating a BGP Neighbor to Support CLNS

In the following example, the router R1, shown in the figure below, in the autonomous system AS65101 is configured to run BGP and activated to support CLNS. Router R1 is the only Level 2 IS-IS router in autonomous system AS65101, and it has only one connection to another autonomous system via router R2 in AS65202. The **no bgp default ipv4-unicast** command is configured on the router to disable the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers. After the NSAP address family configuration mode is enabled with the **address-family nsap** command, the router is configured to advertise the NSAP prefix of 49.0101 to its BGP neighbors and to send NSAP routing information to the BGP neighbor at 10.1.2.2.

```
router bgp 65101
 no bgp default ipv4-unicast
 address-family nsap
  network 49.0101...
  neighbor 10.1.2.2 activate
  exit-address-family
```

Example: Configuring an IS-IS Routing Process

In the following example, R1, shown in the figure below, is configured to run an IS-IS process:

```
router isis osi-as-101
 net 49.0101.1111.1111.1111.1111.00
```

The default IS-IS routing process level is used.

Example: Configuring Interfaces

In the following example, two of the interfaces of the router R2, shown in the figure below, in the autonomous system AS65202 are configured to run CLNS. Ethernet interface 0/1 is connected to the local OSI routing domain and is configured to run IS-IS when the network protocol is CLNS using the **clns router isis** command. The serial interface 2/0 with the local IP address of 10.1.2.2 is connected with an eBGP neighbor and is configured to run CLNS through the **clns enable** command:

```
interface serial 2/0
 ip address 10.1.2.2 255.255.255.0
 clns enable
 no shutdown
!
interface ethernet 0/1
 ip address 10.2.3.1 255.255.255.0
 clns router isis osi-as-202
 no shutdown
```

Example: Advertising Networking Prefixes

In the following example, the router R1, shown in the figure below, is configured to advertise the NSAP prefix of 49.0101 to other routers. The NSAP prefix unique to autonomous system AS65101 is advertised to allow the other autonomous systems to discover the existence of autonomous system AS65101 in the network:

```
router bgp 65101
 no bgp default ipv4-unicast
 neighbor 10.1.2.2 remote-as 64202
 address-family nsap
  network 49.0101...
  neighbor 10.1.2.2 activate
```

Example: Redistributing Routes from BGP into IS-IS

In the following example, the routers R7 and R9, shown in the figure below, in the autonomous system AS65404 are configured to redistribute BGP routes into the IS-IS routing process called osi-as-404. Redistributing the BGP routes allows the Level 2 IS-IS router, R8, to advertise routes to destinations outside the autonomous system AS65404. Without a route map being specified, all BGP routes are redistributed.

Router R7

```
router isis osi-as-404
 net 49.0404.7777.7777.7777.7777.00
 redistribute bgp 65404 clns
```

Router R9

```
router isis osi-as-404
 net 49.0404.9999.9999.9999.9999.00
 redistribute bgp 65404 clns
```

Example: Redistributing Routes from IS-IS into BGP

In the following example, the router R2, shown in the figure below, in the autonomous system AS65202 is configured to redistribute Level 2 CLNS NSAP routes into BGP. A route map is used to permit only routes from within the local autonomous system to be redistributed into BGP. Without a route map being specified, every NSAP route from the CLNS level 2 prefix table is redistributed. The **no bgp default ipv4-unicast** command is configured on the router to disable the default behavior of the BGP routing process exchanging IPv4 addressing information with BGP neighbor routers.

```
clns filter-set internal-routes permit 49.0202...
!
route-map internal-routes-only permit 10
 match clns address internal-routes
!
router isis osi-as-202
 net 49.0202.2222.2222.2222.2222.00
!
router bgp 65202
 no bgp default ipv4-unicast
 address-family nsap
 redistribute isis osi-as-202 clns route-map internal-routes-only
```

Example: Configuring BGP Peer Groups and Route Reflectors

Router R5, shown in the figure above (in the “Generic BGP CLNS Network Topology” section), has only iBGP neighbors and runs IS-IS on both interfaces. To reduce the number of configuration commands, configure R5 as a member of a BGP peer group called ibgp-peers. The peer group is automatically activated under the **address-family nsap** command by configuring the peer group as a route reflector client allowing it to exchange NSAP routing information between group members. The BGP peer group is also configured as a BGP route reflector client to reduce the need for every iBGP router to be linked to each other.

In the following example, the router R5 in the autonomous system AS65303 is configured as a member of a BGP peer group and a BGP route reflector client.

```
router bgp 65303
 no bgp default ipv4-unicast
 neighbor ibgp-peers peer-group
 neighbor ibgp-peers remote-as 65303
 address-family nsap
  neighbor ibgp-peers route-reflector-client
  neighbor 10.4.5.4 peer-group ibgp-peers
  neighbor 10.5.6.6 peer-group ibgp-peers
  exit-address-family
```

Example: Filtering Inbound Routes Based on NSAP Prefixes

In the following example, the router R1, shown in the figure below, in the autonomous system AS65101 is configured to filter inbound routes specified by the default-prefix-only prefix list.

```
clns filter-set default-prefix-only deny 49...
clns filter-set default-prefix-only permit default
!
router isis osi-as-101
 net 49.0101.1111.1111.1111.1111.00
!
router bgp 65101
 no bgp default ipv4-unicast
 neighbor 10.1.2.2 remote-as 64202
 address-family nsap
  network 49.0101.1111.1111.1111.1111.00
  neighbor 10.1.2.2 activate
  neighbor 10.1.2.2 prefix-list default-prefix-only in
```

Example: Filtering Outbound BGP Updates Based on NSAP Prefixes

In the following example, outbound BGP updates are filtered based on NSAP prefixes. This example is configured at Router 7 in the figure below. In this task, a CLNS filter is created with two entries to deny NSAP prefixes starting with 49.0404 and to permit all other NSAP prefixes starting with 49. A BGP peer group is created and the filter is applied to outbound BGP updates for the neighbor that is a member of the peer group.

```
clns filter-set routes0404 deny 49.0404...
clns filter-set routes0404 permit 49...
!
router bgp 65404
 no bgp default ipv4-unicast
 neighbor ebgp-peers remote-as 65303
 address-family nsap
   neighbor ebgp-peers prefix-list routes0404 out
   neighbor 10.6.7.8 peer-group ebgp-peers
```

Example: Originating a Default Route and Outbound Route Filtering

In the figure below, autonomous system AS65101 is connected to only one other autonomous system, AS65202. Router R2 in AS65202 provides the connectivity to the rest of the network for autonomous system AS65101 by sending a default route to R1. Any packets from Level 1 routers within autonomous system AS65101 with destination NSAP addresses outside the local Level 1 network are sent to R1, the nearest Level 2 router. Router R1 forwards the packets to router R2 using the default route.

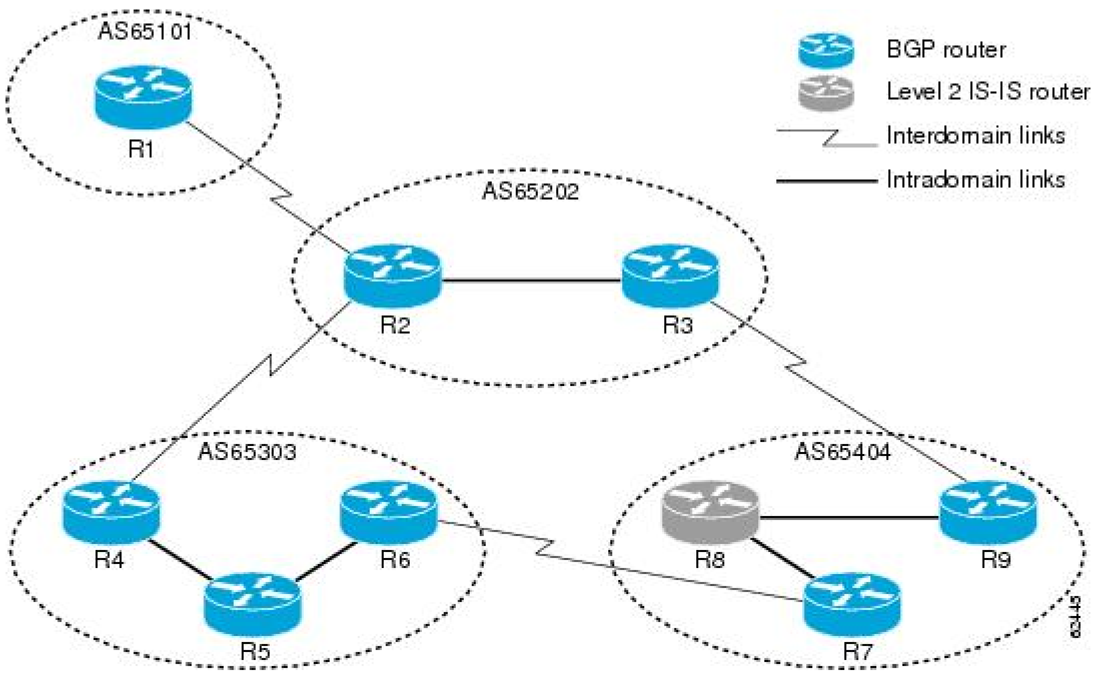
In the following example, the router R2, shown in the figure below, in the autonomous system AS65202 is configured to generate a default route for router R1 in the autonomous system AS65101, and an outbound filter is created to send only the default route NSAP addressing information in the BGP update messages to router R1.

```
clns filter-set default-prefix-only deny 49...
clns filter-set default-prefix-only permit default
!
router bgp 65202
 no bgp default ipv4-unicast
 neighbor 10.1.2.1 remote-as 64101
 address-family nsap
  network 49.0202...
  neighbor 10.1.2.1 activate
  neighbor 10.1.2.1 default-originate
  neighbor 10.1.2.1 prefix-list default-prefix-only out
```

Example: Implementing MP-BGP Support for CLNS

The figure below shows a generic BGP CLNS network containing nine routers that are grouped into four different autonomous systems (in BGP terminology) or routing domains (in OSI terminology). This section contains complete configurations for all routers shown in the figure below.

Figure 19: Components in a Generic BGP CLNS Network



If you need more details about commands used in the following examples, see the configuration tasks earlier in this document and the documents listed in the Additional References, on page 247.

Autonomous System AS65101

Router 1

```
clns filter-set default-prefix-only deny 49...
clns filter-set default-prefix-only permit default
!
router isis osi-as-101
 net 49.0101.1111.1111.1111.1111.00
!
router bgp 65101
 no bgp default ipv4-unicast
 neighbor 10.1.2.2 remote-as 65202
 address-family nsap
  neighbor 10.1.2.2 activate
  neighbor 10.1.2.2 prefix-list default-prefix-only in
  network 49.0101...
  exit-address-family
!
interface serial 2/0
 ip address 10.1.2.1 255.255.255.0
 clns enable
 no shutdown
```

Autonomous System AS65202

Router 2

```
clns filter-set default-prefix-only deny 49...
clns filter-set default-prefix-only permit default
!
clns filter-set internal-routes permit 49.0202...
!
route-map internal-routes-only permit 10
  match clns address internal-routes
!
router isis osi-as-202
  net 49.0202.2222.2222.2222.2222.00
!
router bgp 65202
  no bgp default ipv4-unicast
  neighbor 10.1.2.1 remote-as 65101
  neighbor 10.2.3.3 remote-as 65202
  neighbor 10.2.4.4 remote-as 65303
  address-family nsap
    neighbor 10.1.2.1 activate
    neighbor 10.2.3.3 activate
    neighbor 10.2.4.4 activate
    redistribute isis osi-as-202 clns route-map internal-routes-only
    neighbor 10.1.2.1 default-originate
    neighbor 10.1.2.1 prefix-list default-prefix-only out
    exit-address-family
!
interface ethernet 0/1
  ip address 10.2.3.2 255.255.255.0
  clns router isis osi-as-202
  no shutdown
!
interface serial 2/0
  ip address 10.1.2.2 255.255.255.0
  clns enable
  no shutdown
!
interface serial 2/2
  ip address 10.2.4.2 255.255.255.0
  clns enable
  no shutdown
```

Router 3

```
clns filter-set internal-routes permit 49.0202...
!
route-map internal-routes-only permit 10
  match clns address internal-routes
!
router isis osi-as-202
 net 49.0202.3333.3333.3333.3333.00
!
router bgp 65202
 no bgp default ipv4-unicast
 neighbor 10.2.3.2 remote-as 65202
 neighbor 10.3.9.9 remote-as 65404
 address-family nsap
  neighbor 10.2.3.2 activate
  neighbor 10.3.9.9 activate
```

```
  redistribute isis osi-as-202 clns route-map internal-routes-only
  exit-address-family
!
interface ethernet 0/1
 ip address 10.2.3.3 255.255.255.0
 clns router isis osi-as-202
 no shutdown
!
interface serial 2/2
 ip address 10.3.9.3 255.255.255.0
 clns enable
 no shutdown
```

Autonomous System AS65303

Router 4

```
router isis osi-as-303
 net 49.0303.4444.4444.4444.4444.00
!
router bgp 65303
 no bgp default ipv4-unicast
 neighbor 10.2.4.2 remote-as 65202
 neighbor 10.4.5.5 remote-as 65303
 address-family nsap
  no synchronization
  neighbor 10.2.4.2 activate
  neighbor 10.4.5.5 activate
  network 49.0303...
  exit-address-family
!
interface ethernet 0/2
 ip address 10.4.5.4 255.255.255.0
 clns router isis osi-as-303
 no shutdown
!
interface serial 2/3
 ip address 10.2.4.4 255.255.255.0
 clns enable
 no shutdown
```

Router 5

```
router isis osi-as-303
 net 49.0303.5555.5555.5555.5555.00
!
router bgp 65303
 no bgp default ipv4-unicast
 neighbor ibgp-peers peer-group
 neighbor ibgp-peers remote-as 65303
 address-family nsap
  no synchronization
  neighbor ibgp-peers route-reflector-client
  neighbor 10.4.5.4 peer-group ibgp-peers
  neighbor 10.5.6.6 peer-group ibgp-peers
  exit-address-family
!
interface ethernet 0/2
 ip address 10.4.5.5 255.255.255.0
 clns router isis osi-as-303
```

```
 no shutdown
!
interface ethernet 0/3
 ip address 10.5.6.5 255.255.255.0
 clns router isis osi-as-303
 no shutdown
```

Router 6

```
router isis osi-as-303
 net 49.0303.6666.6666.6666.6666.00
!
router bgp 65303
 no bgp default ipv4-unicast
 neighbor 10.5.6.5 remote-as 65303
 neighbor 10.6.7.7 remote-as 65404
 address-family nsap
  no synchronization
  neighbor 10.5.6.5 activate
  neighbor 10.6.7.7 activate
  network 49.0303...
!
interface ethernet 0/3
 ip address 10.5.6.6 255.255.255.0
 clns router isis osi-as-303
 no shutdown
!
interface serial 2/2
 ip address 10.6.7.6 255.255.255.0
 clns enable
 no shutdown
```

Autonomous System AS65404

Router 7

```
clns filter-set external-routes deny 49.0404...
clns filter-set external-routes permit 49...
!
route-map noexport permit 10
  match clns address external-routes
  set community noexport
!
router isis osi-as-404
 net 49.0404.7777.7777.7777.7777.00
 redistribute bgp 404 clns
!
router bgp 65404
 no bgp default ipv4-unicast
 neighbor 10.6.7.6 remote-as 65303
 neighbor 10.8.9.9 remote-as 65404
 address-family nsap
  neighbor 10.6.7.6 activate
  neighbor 10.8.9.9 activate
  neighbor 10.8.9.9 send-community
  neighbor 10.8.9.9 route-map noexport out
  network 49.0404...
!
interface ethernet 1/0
 ip address 10.7.8.7 255.255.255.0
```

```
 clns router isis osi-as-404
 ip router isis osi-as-404
 no shutdown
!
interface serial 2/3
 ip address 10.6.7.7 255.255.255.0
 clns enable
 no shutdown
```

Router 8

```
router isis osi-as-404
 net 49.0404.8888.8888.8888.8888.00
!
interface ethernet 1/0
 ip address 10.7.8.8 255.255.255.0
 clns router isis osi-as-404
 ip router isis osi-as-404
 no shutdown
!
interface ethernet 1/1
 ip address 10.8.9.8 255.255.255.0
 clns router isis osi-as-404
 ip router isis osi-as-404
 no shutdown
```

Router 9

```
clns filter-set external-routes deny 49.0404...
clns filter-set external-routes permit 49...
!
route-map noexport permit 10
  match clns address external-routes
  set community noexport
!
router isis osi-as-404
 net 49.0404.9999.9999.9999.9999.00
 redistribute bgp 404 clns
!
router bgp 65404
 no bgp default ipv4-unicast
 neighbor 10.3.9.3 remote-as 65202
 neighbor 10.7.8.7 remote-as 65404
 address-family nsap
  network 49.0404...
  neighbor 10.3.9.3 activate
  neighbor 10.7.8.7 activate
  neighbor 10.7.8.7 send-community
  neighbor 10.7.8.7 route-map noexport out
!
interface serial 2/3
 ip address 10.3.9.9 255.255.255.0
 clns enable
 no shutdown
!
interface ethernet 1/1
 ip address 10.8.9.9 255.255.255.0
 clns router isis osi-as-404
 ip router isis osi-as-404
 no shutdown
```

Additional References

Related Documents

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

Standards

Standard	Title
ISO/IEC 8473	*ISO CLNP: Connectionless Network Protocol (ISO-IP)*. Protocol for providing the connectionless-mode network service.
ISO/IEC 9542	*End System to Intermediate System Protocol (ESIS)*. End system to Intermediate system routing exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473).
ISO/IEC 10589	*IS-IS, Intermediate System-to-Intermediate System*. Intermediate system to Intermediate system intradomain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473).

MIBs

MIB	MIBs Link
None.	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 1700	*Assigned Numbers*
RFC 1771	*A Border Gateway Protocol 4 (BGP-4)*
RFC 1997	*BGP Communities Attribute*
RFC 2042	*Registering New BGP Attribute Types*
RFC 2439	*BGP Route Flap Dampening*
RFC 2842	*Capabilities Advertisement with BGP-4*
RFC 2858	*Multiprotocol Extensions for BGP-4*

RFC	Title
RFC 2918	*Route Refresh Capability for BGP-4*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register on Cisco.com.	http://www.cisco.com/techsupport

Feature Information for Configuring MP-BGP Support for CLNS

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 MP-BGP Support for CLNS

Feature Name	Releases	Feature Information
Multiprotocol BGP (MP-BGP) Support for CLNS	12.2(8)T 12.2(33)SRB	The Multiprotocol BGP (MP-BGP) Support for CLNS feature provides the ability to scale Connectionless Network Service (CLNS) networks. The multiprotocol extensions of Border Gateway Protocol (BGP) add the ability to interconnect separate Open System Interconnection (OSI) routing domains without merging the routing domains, thus providing the capability to build very large OSI networks. In Release 12.2(8)T, this feature was introduced on the following platforms: • Cisco 2600 series • Cisco 3600 series • Cisco 7100 series • Cisco 7200 series • Cisco 7500 series • Cisco uBR7200 series In Release 12.2(33)SRB, this feature was introduced on the Cisco 7600 Series. The following commands were introduced or modified by this feature: **address-family nsap**, **clear bgp nsap**, **clear bgp nsap dampening**, **clear bgp nsap external**, **clear bgp nsap flap-statistics**, **clear bgp nsap peer-group**, **debug bgp nsap**, **debug bgp nsap dampening**, **debug bgp nsap updates**, **neighbor prefix-list**, **network (BGP and multiprotocol BGP)**, **redistribute (BGP to ISO ISIS)**, **redistribute (ISO ISIS to BGP)**, **show bgp nsap**, **show bgp nsap community**, **show bgp nsap community-list**, **show bgp nsap dampened-paths**, **show bgp nsap filter-list**, **show bgp nsap flap-statistics**, **show bgp nsap inconsistent-as**, **show bgp nsap neighbors**, **show bgp nsap paths**, **show bgp nsap quote-regexp**, **show bgp nsap regexp**, **show bgp nsap summary.**

Glossary

address family —A group of network protocols that share a common format of network address. Address families are defined by RFC 1700.

AS —autonomous system. An IP term to describe a routing domain that has its own independent routing policy and is administered by a single authority. Equivalent to the OSI term “routing domain.”

BGP —Border Gateway Protocol. Interdomain routing protocol that exchanges reachability information with other BGP systems.

CLNS —Connectionless Network Service. An OSI network-layer protocol.

CMIP —Common Management Information Protocol. In OSI, a network management protocol created and standardized by ISO for the monitoring and control of heterogeneous networks.

DCC —data communications channel.

DCN —data communications network.

ES-IS —End System-to-Intermediate System. OSI protocol that defines how end systems (hosts) announce themselves to intermediate systems (routers).

FTAM —File Transfer, Access, and Management. In OSI, an application-layer protocol developed for network file exchange and management between diverse types of computers.

IGP —Interior Gateway Protocol. Internet protocol used to exchange routing information within an autonomous system.

IGRP —Interior Gateway Routing Protocol. A proprietary Cisco protocol developed to address the issues associated with routing in large, heterogeneous networks.

IS —intermediate system. Routing node in an OSI network.

IS-IS —Intermediate System-to-Intermediate System. OSI link-state hierarchical routing protocol based on DECnet Phase V routing, where routers exchange routing information based on a single metric, to determine network topology.

ISO —International Organization for Standardization. International organization that is responsible for a wide range of standards, including those relevant to networking. ISO developed the Open System Interconnection (OSI) reference model, a popular networking reference model.

NSAP address —network service access point address. The network address format used by OSI networks.

OSI —Open System Interconnection. International standardization program created by ISO and ITU-T to develop standards for data networking that facilitate multivendor equipment interoperability.

routing domain —The OSI term that is equivalent to autonomous system for BGP.

SDH —Synchronous Digital Hierarchy. Standard that defines a set of rate and format standards that are sent using optical signals over fiber.

SONET —Synchronous Optical Network. High-speed synchronous network specification designed to run on optical fiber.

CHAPTER 10

Connecting to a Service Provider Using External BGP

This module describes configuration tasks that will enable your Border Gateway Protocol (BGP) network to access peer devices in external networks such as those from Internet service providers (ISPs). BGP is an interdomain routing protocol that is designed to provide loop-free routing between organizations. External BGP (eBGP) peering sessions are configured to allow peers from different autonomous systems to exchange routing updates. Tasks to help manage the traffic that is flowing inbound and outbound are described, as are tasks to configure BGP policies to filter the traffic. Multihoming techniques that provide redundancy for connections to a service provider are also described.

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 Connecting to a Service Provider Using External BGP

- Before connecting to a service provider you need to understand how to configure the basic BGP process and peers. See the “Cisco BGP Overview” and “Configuring a Basic BGP Network” modules for more details.
- The tasks and concepts in this chapter will help you configure BGP features that would be useful if you are connecting your network to a service provider. For each connection to the Internet, you must have an assigned autonomous system number from the Internet Assigned Numbers Authority (IANA).

Restrictions for Connecting to a Service Provider Using External BGP

- A router that runs Cisco IOS 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 address family configurations.
- Policy lists are not supported in versions of Cisco IOS software prior to Cisco IOS Release 12.0(22)S and 12.2(15)T. Reloading a router that is running an older version of Cisco IOS software may cause some routing policy configurations to be lost.

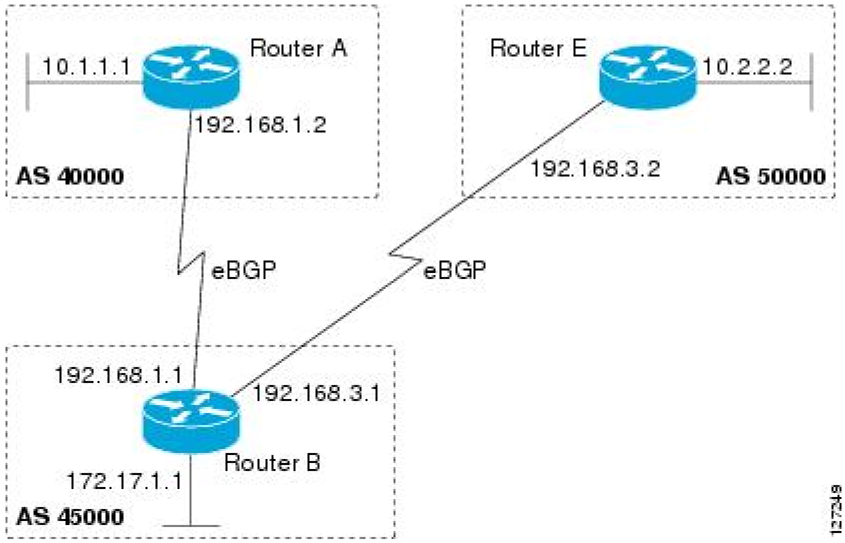
Information About Connecting to a Service Provider Using External BGP

External BGP Peering

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 and it uses TCP (port 179) as the transport protocol. The destination TCP port is assigned 179, and the local port is assigned a random port number. Cisco IOS software supports BGP version 4, which has been used by 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.

External BGP peering sessions are configured to allow BGP peers from different autonomous systems to exchange routing updates. By design, a BGP routing process expects eBGP peers to be directly connected, for example, over a WAN connection. However, there are many real-world scenarios where this rule would prevent routing from occurring. Peering sessions for multihop neighbors are configured with the **neighbor ebgp-multihop** command. The figure below shows simple eBGP peering between three routers. Router B peers with Router A and Router E. In the figure below, the **neighbor ebgp-multihop** command could be used to establish peering between Router A and Router E although this is a very simple network design. BGP forwards information about the next hop in the network using the NEXT_HOP attribute, which is set to the IP address of the interface that advertises a route in an eBGP peering session by default. The source interface can be a physical interface or a loopback interface.

Figure 20: BGP Peers in Different Autonomous Systems



Loopback interfaces are preferred for establishing eBGP peering sessions because loopback interfaces are less susceptible to interface flapping. Interfaces on networking devices can fail, and they can also be taken out of service for maintenance. When an interface is administratively brought up or down, due to failure or maintenance, it is referred to as a flap. Loopback interfaces provide a stable source interface to ensure that the IP address assigned to the interface is always reachable as long as the IP routing protocols continue to advertise the subnet assigned to the loopback interface. Loopback interfaces allow you to conserve address space by configuring a single address with /32 bit mask. Before a loopback interface is configured for an eBGP peering session, you must configure the **neighbor update-source** command and specify the loopback interface. With this configuration, the loopback interface becomes the source interface and its IP address is advertised as the next hop for routes that are advertised through this loopback. If loopback interfaces are used to connect single-hop eBGP peers, you must configure the **neighbor disable-connected-check** command before you can establish the eBGP peering session.

Connecting to external networks enables traffic from your network to be forwarded to other networks and across the Internet. Traffic will also be flowing into, and possibly through, your network. BGP contains various techniques to influence how the traffic flows into and out of your network, and to create BGP policies that filter the traffic, inbound and outbound. To influence the traffic flow, BGP uses certain BGP attributes that can be included in update messages or used by the BGP routing algorithm. BGP policies to filter traffic also use some of the BGP attributes with route maps, access lists including AS-path access lists, filter lists, policy lists, and distribute lists. Managing your external connections may involve multihoming techniques where there is more than one connection to an ISP or connections to more than one ISP for backup or performance purposes. Tagging BGP routes with different community attributes across autonomous system or physical boundaries can prevent the need to configure long lists of individual permit or deny statements.

BGP Autonomous System Number Formats

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

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

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

Asdot Only Autonomous System Number Formatting

In Cisco IOS 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 19: 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 20: 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 21: 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/.

BGP Attributes

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 various attributes that are used in BGP best-path analysis. Cisco IOS software provides the ability to influence BGP path selection by altering these attributes via the command-line interface (CLI). BGP path selection can also be influenced through standard BGP policy configuration.

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, then the oldest paths are selected as multipaths.

BGP can include path attribute information in update messages. BGP attributes describe the characteristic of the route, and the software uses these attributes to help make decisions about which routes to advertise. Some of this attribute information can be configured at a BGP-speaking networking device. There are some mandatory attributes that are always included in the update message and some discretionary attributes. The following BGP attributes can be configured:

- AS_Path
- Community
- Local_Pref
- Multi_Exit_Discriminator (MED)
- Next_Hop
- Origin

AS_Path

This attribute contains a list or set of the autonomous system numbers through which routing information has passed. The BGP speaker adds its own autonomous system number to the list when it forwards the update message to external peers.

Community

BGP communities are used to group networking devices that share common properties, regardless of network, autonomous system, or any physical boundaries. In large networks applying a common routing policy through prefix lists or access lists requires individual peer statements on each networking device. Using the BGP community attribute BGP neighbors, with common routing policies, can implement inbound or outbound route filters based on the community tag rather than consult large lists of individual permit or deny statements.

Local_Pref

Within an autonomous system, the Local_Pref attribute is included in all update messages between BGP peers. If there are several paths to the same destination, the local preference attribute with the highest value indicates the preferred outbound path from the local autonomous system. The highest ranking route is advertised to internal peers. The Local_Pref value is not forwarded to external peers.

Multi_Exit_Discriminator

The MED attribute indicates (to an external peer) a preferred path into an autonomous system. If there are multiple entry points into an autonomous system, the MED can be used to influence another autonomous system to choose one particular entry point. A metric is assigned where a lower MED metric is preferred by the software over a higher MED metric. The MED metric is exchanged between autonomous systems, but after a MED is forwarded into an autonomous system, the MED metric is reset to the default value of 0. When an update is sent to an internal BGP (iBGP) peer, the MED is passed along without any change, allowing all the peers in the same autonomous system to make a consistent path selection.

By default, a router will compare the MED attribute for paths only from BGP peers that reside in the same autonomous system. The **bgp always-compare-med** command can be configured to allow the router to compare metrics from peers in different autonomous systems.

Note The Internet Engineering Task Force (IETF) decision regarding BGP MED assigns a value of infinity to the missing MED, making the route that lacks the MED variable the least preferred. The default behavior of BGP routers that run Cisco software is to treat routes without the MED attribute as having a MED of 0, making the route that lacks the MED variable the most preferred. To configure the router to conform to the IETF standard, use the **bgp bestpath med missing-as-worst** router configuration command.

Next_Hop

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

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

Origin

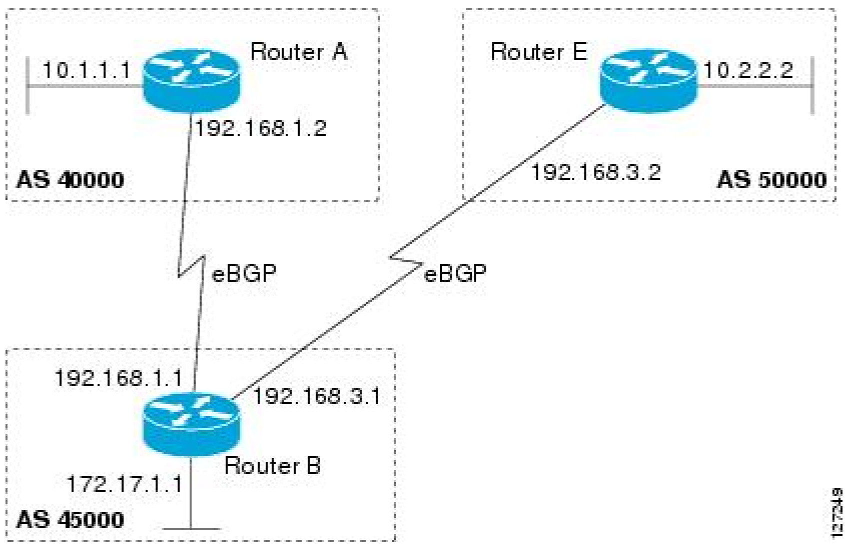
This attribute indicates how the route was included in a BGP routing table. In Cisco software, a route defined using the BGP **network** command is given an origin code of Interior Gateway Protocol (IGP). Routes distributed from an Exterior Gateway Protocol (EGP) are coded with an origin of EGP, and routes redistributed from other protocols are defined as Incomplete. BGP decision policy for origin prefers IGP over EGP, and then EGP over Incomplete.

Multihoming

Multihoming is defined as connecting an autonomous system with more than one service provider. If you have any reliability issues with one service provider, then you have a backup connection. Performance issues can also be addressed by multihoming because better paths to the destination network can be utilized.

Unless you are a service provider, you must plan your routing configuration carefully to avoid Internet traffic traveling through your autonomous system and consuming all your bandwidth. The figure below shows that autonomous system 45000 is multihomed to autonomous system 40000 and autonomous system 50000. Assuming autonomous system 45000 is not a service provider, then several techniques such as load balancing or some form of routing policy must be configured to allow traffic from autonomous system 45000 to reach either autonomous system 40000 or autonomous system 50000 but not allow much, if any, transit traffic.

Figure 21: Multihoming Topology



MED Attribute

Configuring the MED attribute is another method that BGP can use to influence the choice of paths into another autonomous system. The MED attribute indicates (to an external peer) a preferred path into an autonomous system. If there are multiple entry points into 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.

Transit Versus Nontransit Traffic

Most of the traffic within an autonomous system contains a source or destination IP address residing within the autonomous system, and this traffic is referred to as nontransit (or local) traffic. Other traffic is defined as transit traffic. As traffic across the Internet increases, controlling transit traffic becomes more important.

A service provider is considered to be a transit autonomous system and must provide connectivity to all other transit providers. In reality, few service providers actually have enough bandwidth to allow all transit traffic, and most service providers have to purchase such connectivity from Tier 1 service providers.

An autonomous system that does not usually allow transit traffic is called a stub autonomous system and will link to the Internet through one service provider.

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 22: 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

Note In Cisco IOS Releases 12.0(22)S, 12.2(15)T, 12.2(18)S, and later releases, the maximum number of autonomous system access lists that can be configured with the **ip as-path access-list** command is increased from 199 to 500.

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 IOS 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.
```

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.

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.

Extended Communities

Extended community attributes are used to configure, filter, and identify routes for virtual routing and forwarding (VRF) instances and Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs). All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers. All regular expression configuration options are supported. The route target (RT) and site of origin (SoO) extended community attributes are supported by the standard range of extended community lists.

Route Target Extended Community Attribute

The RT extended community attribute is configured with the **rt** keyword of the **ip extcommunity-list** command. This attribute is used to identify a set of sites and VRFs that may receive routes that are tagged with the configured route target. Configuring the route target extended community attribute with a route allows that route to be placed in the per-site forwarding tables that are used for routing traffic that is received from corresponding sites.

Site of Origin Extended Community Attribute

The SoO extended community attribute is configured with the **soo** keyword of the **ip extcommunity-list** command. This attribute uniquely identifies the site from which the provider edge (PE) router learned the route. All routes learned from a particular site must be assigned the same SoO extended community attribute, regardless if a site is connected to a single PE router or multiple PE routers. Configuring this attribute prevents routing loops from occurring when a site is multihomed. The SoO extended community attribute is configured on the interface and is propagated into BGP through redistribution. The SoO extended community attribute can be applied to routes that are learned from VRFs. The SoO extended community attribute should not be configured for stub sites or sites that are not multihomed.

IP Extended Community-List Configuration Mode

Named and numbered extended community lists can be configured in IP extended community-list configuration mode. The IP extended community-list configuration mode supports all of the functions that are available in global configuration mode. In addition, the following operations can be performed:

- Configure sequence numbers for extended community list entries.
- Resequence existing sequence numbers for extended community list entries.
- Configure an extended community list to use default values.

Default Sequence Numbering

Extended community list entries start with the number 10 and increment by 10 for each subsequent entry when no sequence number is specified, when default behavior is configured, and when an extended community list is resequenced without specifying the first entry number or the increment range for subsequent entries.

Resequencing Extended Community Lists

Extended community-list entries are sequenced and resequenced on a per-extended community list basis. The **resequence** command can be used without any arguments to set all entries in a list to default sequence numbering. The **resequence** command also allows the sequence number of the first entry and increment range to be set for each subsequent entry. The range of configurable sequence numbers is from 1 to 2147483647.

Extended Community Lists

Extended community attributes are used to configure, filter, and identify routes for VRF instances and MPLS VPNs. The **ip extcommunity-list** command is used to configure named or numbered extended community lists. All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers.

Administrative Distance

Administrative distance is a measure of the preference of different routing protocols. BGP has a **distance bgp** command that allows you to set different administrative distances for three route types: external, internal, and local. BGP, like other protocols, prefers the route with the lowest administrative distance.

BGP Route Map Policy Lists

BGP route map policy lists allow a network operator to group route map match clauses into named lists called policy lists. A policy list functions like a macro. When a policy list is referenced in a route map, all of the match clauses are evaluated and processed as if they had been configured directly in the route map. This enhancement simplifies the configuration of BGP routing policy in medium-size and large networks because a network operator can preconfigure policy lists with groups of match clauses and then reference these policy lists within different route maps. The network operator no longer needs to manually reconfigure each recurring group of match clauses that occur in multiple route map entries.

A policy lists functions like a macro when it is configured in a route map and has the following capabilities and characteristics:

- When a policy list is referenced within a route map, all the match statements within the policy list are evaluated and processed.
- Two or more policy lists can be configured with a route map. Policy lists can be configured within a route map to be evaluated with AND or OR semantics.
- Policy lists can coexist with any other preexisting match and set statements that are configured within the same route map but outside of the policy lists.
- When multiple policy lists perform matching within a route map entry, all policy lists match on the incoming attribute only.

Policy lists support only match clauses and do not support set clauses. Policy lists can be configured for all applications of route maps, including redistribution, and can also coexist, within the same route map entry, with match and set clauses that are configured separately from the policy lists.

Note Policy lists are supported only by BGP and are not supported by other IP routing protocols.

How to Connect to a Service Provider Using External BGP

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:

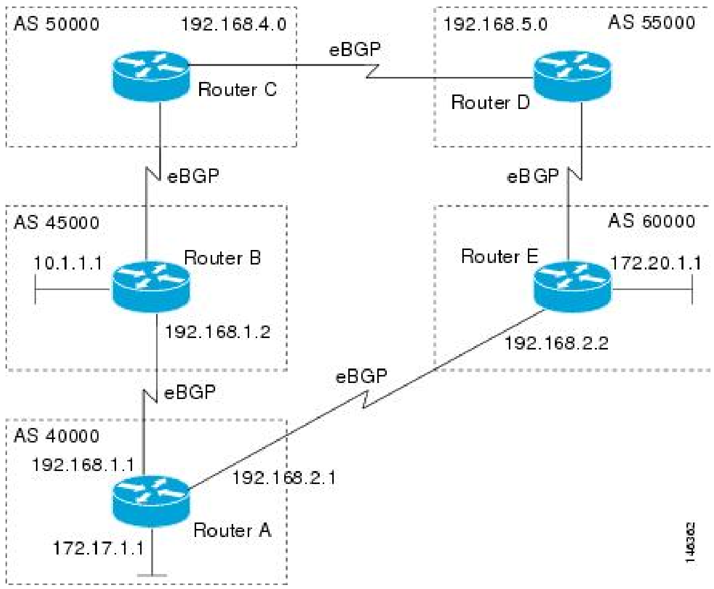
Influencing Inbound Path Selection by Modifying the AS_PATH Attribute

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 22: 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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
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.
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:**	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.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 192.168.1.2 route-map PREPEND out`	
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.
Step 14	**show running-config** **Example:** `Device# show running-config`	Displays the running configuration file.

Examples

Router A

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

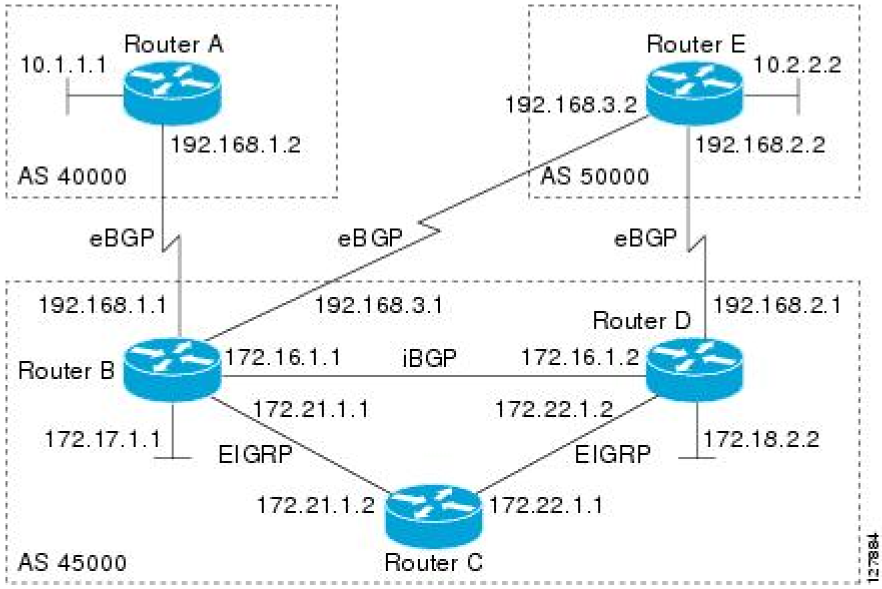
```
Device# show running-config
.
.
.
router bgp 40000
 neighbor 192.168.1.2 remote-as 45000
 !
 address-family ipv4
 neighbor 192.168.1.2 activate
 neighbor 192.168.1.2 route-map PREPEND out
 no auto-summary
 no synchronization
 network 172.17.1.0 mask 255.255.255.0
 exit-address-family
!
route-map 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 23: 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.

	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* | *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.
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.

	Command or Action	Purpose
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
```

Influencing Outbound Path Selection

BGP can be used to influence the choice of paths for outbound traffic from the local autonomous system. This section contains two methods that BGP can use to influence outbound path selection:

- Using the Local_Pref attribute
- Using the BGP outbound route filter (ORF) capability

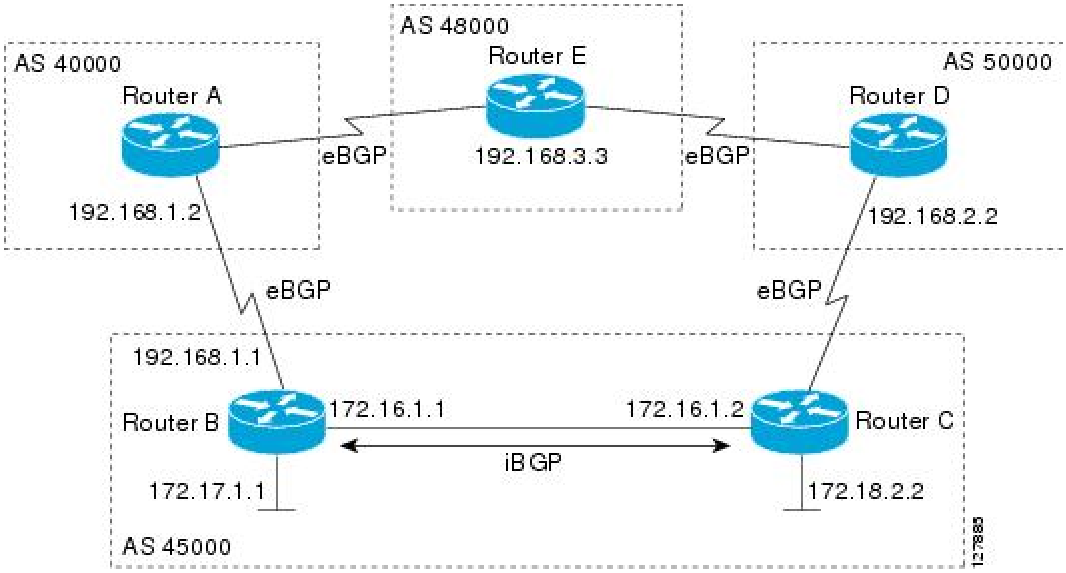
Perform one of the following tasks to influence outbound path selection:

Influencing Outbound Path Selection Using the Local_Pref Attribute

One of the methods to influence outbound path selection is to use the BGP Local-Pref attribute. Perform this task using the local preference attribute to influence outbound path selection. If there are several paths to the same destination the local preference attribute with the highest value indicates the preferred path.

Refer to the figure below for the network topology used in this task. Both Router B and Router C are configured. autonomous system 45000 receives updates for network 192.168.3.0 via autonomous system 40000 and autonomous system 50000. Router B is configured to set the local preference value to 150 for all updates to autonomous system 40000. Router C is configured to set the local preference value for all updates to autonomous system 50000 to 200. After the configuration, local preference information is exchanged within autonomous system 45000. Router B and Router C now see that updates for network 192.168.3.0 have a higher preference value from autonomous system 50000 so all traffic in autonomous system 45000 with a destination network of 192.168.3.0 is sent out via Router C.

Figure 24: Network Topology for Outbound Path Selection



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. **bgp default local-preference** *value*
6. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
7. **network** *network-number* [**mask** *network-mask*][**route-map** *route-map-name*]
8. **neighbor** {*ip-address*| *peer-group-name*} **activate**
9. **end**
10. Repeat Step 1 through Step 9 at Router C but change the IP address of the peer, the autonomous system number, and set the local preference value to 200.
11. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 5	**bgp default local-preference** *value* **Example:** `Router(config-router)# bgp default local-preference 150`	Changes the default local preference value. • In this example, the local preference is changed to 150 for all updates from autonomous system 40000 to autonomous system 45000. • By default, the local preference value is 100.
Step 6	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(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 7	**network** *network-number* [**mask** *network-mask*][**route-map** *route-map-name*] **Example:** `Router(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	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 8	**neighbor** {*ip-address*| *peer-group-name*} **activate** **Example:** `Router(config-router-af)# neighbor 192.168.1.2 activate`	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.

	Command or Action	Purpose
Step 9	**end** **Example:** `Router(config-router-af)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 10	Repeat Step 1 through Step 9 at Router C but change the IP address of the peer, the autonomous system number, and set the local preference value to 200.	--
Step 11	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# show ip bgp 192.168.3.0 255.255.255.0`	Displays the entries in the BGP routing table. • Enter this command at both Router B and Router C and note the Local_Pref value. The route with the highest preference value will be the preferred route to network 192.168.3.0. **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*.

Filtering Outbound BGP Route Prefixes

Perform this task to use BGP prefix-based outbound route filtering to influence outbound path selection.

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. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **ebgp-multihop** [*hop-count*]
7. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]

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:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Router(config)# ip prefix-list 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:** `Router(config)# router bgp 100`	Enters router configuration mode, and creates a BGP routing process.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# 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 6	**neighbor** *ip-address* **ebgp-multihop** [*hop-count*] **Example:**	Accepts or initiates BGP connections to external peers residing on networks that are not directly connected.

	Command or Action	Purpose
	`Router(config-router)# neighbor 10.1.1.1 ebgp-multihop`	
Step 7	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 8	**neighbor** *ip-address* **capability orf prefix-list** [**send** | **receive** | **both**] **Example:** `Router(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**} **Example:** `Router(config-router-af)# neighbor 10.1.1.1 prefix-list FILTER in`	Applies an inbound prefix-list filter to prevent distribution of BGP neighbor information. • 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.

	Command or Action	Purpose
Step 10	**end** **Example:** `Router(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:** `Router# 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.

Configuring BGP Peering with ISPs

BGP was developed as an interdomain routing protocol and connecting to ISPs is one of the main functions of BGP. Depending on the size of your network and the purpose of your business, there are many different ways to connect to your ISP. Multihoming to one or more ISPs provides redundancy in case an external link to an ISP fails. This section introduces some optional tasks that can be used to connect to a service provider using multihoming techniques. Smaller companies may use just one ISP but require a backup route to the ISP. Larger companies may have access to two ISPs, using one of the connections as a backup, or may need to configure a transit autonomous system.

Perform one of the following optional tasks to connect to one or more ISPs:

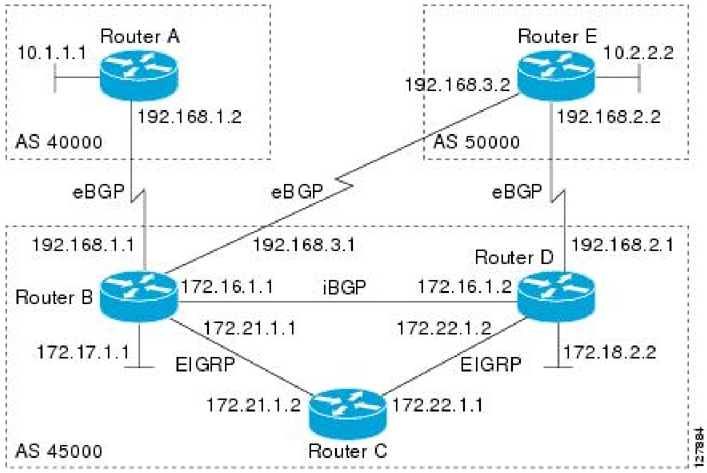
Configuring Multihoming with Two ISPs

Perform this task to configure your network to access two ISPs where one ISP is the preferred route and the second ISP is a backup route. In the figure below Router B in autonomous system 45000 has BGP peers in two ISPs, autonomous system 40000 and autonomous system 50000. Using this task, Router B will be configured to prefer the route to the BGP peer at Router A in autonomous system 40000.

All routes learned from this neighbor will have an assigned weight. The route with the highest weight will be chosen as the preferred route when multiple routes are available to a particular network.

Note The weights assigned with the **set weight** route-map configuration command override the weights assigned using the **neighbor weight** command.

Figure 25: Multihoming with Two ISPs



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*]
7. **neighbor** {*ip-address* | *peer-group-name*} **weight** *number*
8. **exit**
9. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
10. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
11. **neighbor** {*ip-address* | *peer-group-name*} **weight** *number*
12. **end**
13. **clear ip bgp** {* | *ip-address* | *peer-group-name*} [**soft** [**in** | **out**]]
14. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode, and creates a BGP routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 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*] **Example:** `Router(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.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **weight** *number* **Example:** `Router(config-router-af)# neighbor 192.168.1.2 weight 150`	Assigns a weight to a BGP peer connection. • In this example, the weight attribute for routes received from the BGP peer 192.168.1.2 is set to 150.
Step 8	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(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 10	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 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 11	**neighbor** {*ip-address* | *peer-group-name*} **weight** *number* **Example:** `Router(config-router-af)# neighbor 192.168.3.2 weight 100`	Assigns a weight to a BGP peer connection. • In this example, the weight attribute for routes received from the BGP peer 192.168.3.2 is set to 100.
Step 12	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.
Step 13	**clear ip bgp** {***** | *ip-address* | *peer-group-name*} [**soft** [**in** | **out**]] **Example:** `Router# clear ip bgp *`	(Optional) Clears BGP outbound route filters and initiates an outbound soft reset. A single neighbor or all neighbors can be specified.
Step 14	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# show ip bgp`	Displays the entries in the BGP routing table. • Enter this command at Router B to see the weight attribute for each route to a BGP peer. The route with the highest weight attribute will be the preferred route to network 172.17.1.0. **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 example shows the BGP routing table at Router B with the weight attributes assigned to routes. The route through 192.168.1.2 (Router A in the figure above) has the highest weight attribute and will be the preferred route to network 10.3.0.0, wherein the network 10.3.0.0 is accessible through Router A and Router E. If this route (through Router B) fails for some reason, the route through 192.168.3.2 (Router E) will be used to reach network 10.3.0.0. This way, redundancy is provided for reaching Router B.

```
BGP table version is 8, 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             150      40000 i
*> 10.2.2.0/24         192.168.3.2          0             100      50000 i

*> 10.3.0.0/16         192.168.1.2          0             150      40000 i
*                      192.168.3.2          0             100      50000 i

*> 172.17.1.0/24       0.0.0.0              0             32768          i
```

Multihoming with a Single ISP

Perform this task to configure your network to access one of two connections to a single ISP, where one of the connections is the preferred route and the second connection is a backup route. In the figure above Router E in autonomous system 50000 has two BGP peers in a single autonomous system, autonomous system 45000. Using this task, autonomous system 50000 does not learn any routes from autonomous system 45000 and is sending its own routes using BGP. This task is configured at Router E in the figure above and covers three features about multihoming to a single ISP:

- Outbound traffic—Router E will forward default routes and traffic to autonomous system 45000 with Router B as the primary link and Router D as the backup link. Static routes are configured to both Router B and Router D with a lower distance configured for the link to Router B.
- Inbound traffic—Inbound traffic from autonomous system 45000 is configured to be sent from Router B unless the link fails when the backup route is to send traffic from Router D. To achieve this, outbound filters are set using the MED metric.
- Prevention of transit traffic—A route map is configured at Router E in autonomous system 50000 to block all incoming BGP routing updates to prevent autonomous system 50000 from receiving transit traffic from the ISP in autonomous system 45000.

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. Repeat Step 7 to apply another route map to the neighbor specified in Step 7.
9. **exit**
10. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
11. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
12. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
13. Repeat Step 10 to apply another route map to the neighbor specified in Step 10.
14. **exit**
15. **exit**
16. **ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]
17. Repeat Step 14 to establish another static route.
18. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
19. **set metric** *value*
20. **exit**
21. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
22. **set metric** *value*
23. **exit**
24. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
25. **end**
26. **show ip route** [*ip-address*] [*mask*] [**longer-prefixes**]
27. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:**	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.

	Command or Action	Purpose
	`Router(config-router)# neighbor 192.168.2.1 remote-as 45000`	• In this example, the BGP peer at Router D is added to the BGP routing table.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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:** `Router(config-router-af)# network 10.2.2.0 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.

Step 7	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
	Example:
	`Router(config-router-af)# neighbor 192.168.2.1 route-map BLOCK in`
	Example:
	`Router(config-router-af)# neighbor 192.168.2.1 route-map SETMETRIC1 out`
		Applies a route map to incoming or outgoing routes.
		• In the first example, the route map named BLOCK is applied to inbound routes at Router E.
		• In the second example, the route map named SETMETRIC1 is applied to outbound routes to Router D.
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 8	Repeat Step 7 to apply another route map to the neighbor specified in Step 7.	--
Step 9	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.

	Command or Action	Purpose
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.3.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • In this example, the BGP peer at Router D is added to the BGP routing table.
Step 11	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 12	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
	Example:
	`Router(config-router-af)# neighbor 192.168.3.1 route-map BLOCK in`
	Example:
	`Router(config-router-af)# neighbor 192.168.3.1 route-map SETMETRIC2 out`
		Applies a route map to incoming or outgoing routes.
		• In the first example, the route map named BLOCK is applied to inbound routes at Router E.
		• In the second example, the route map named SETMETRIC2 is applied to outbound routes to Router D.
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 13	Repeat Step 10 to apply another route map to the neighbor specified in Step 10.	--
Step 14	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 15	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.

	Command or Action	Purpose

Step 16	**ip route** *prefix mask* {*ip-address* | *interface-type interface-number* [*ip-address*]} [*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*]
	Example:
	`Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.2.1 50`
	Example:
	`Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.2.1 50`
	Example:
	`and`
	Example:
	`Router(config)# ip route 0.0.0.0 0.0.0.0 192.168.3.1 40`
		Establishes a static route.
		• In the first example, a static route to BGP peer 192.168.2.1 is established and given an administrative distance of 50.
		• In the second example, a static route to BGP peer 192.168.3.1 is established and given an administrative distance of 40. The lower administrative distance makes this route via Router B the preferred route.
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 17	Repeat Step 14 to establish another static route.	--
Step 18	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map SETMETRIC1 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named SETMETRIC1 is created.
Step 19	**set metric** *value* **Example:** `Router(config-route-map)# set metric 100`	Sets the MED metric value.
Step 20	**exit** **Example:** `Router(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 21	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map SETMETRIC2 permit 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named SETMETRIC2 is created.
Step 22	**set metric** *value* **Example:** `Router(config-route-map)# set metric 50`	Sets the MED metric value.

	Command or Action	Purpose
Step 23	**exit** **Example:** `Router(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 24	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map BLOCK deny 10`	Configures a route map and enters route map configuration mode. • In this example, a route map named BLOCK is created to block all incoming routes from autonomous system 45000.
Step 25	**end** **Example:** `Router(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 26	**show ip route** [*ip-address*] [*mask*] [**longer-prefixes**] **Example:** `Router# show ip route`	(Optional) Displays route information from the routing tables. • Use this command at Router E in the figure above after Router B and Router D have received update information containing the MED metric from Router E. • 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 27	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# 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 after Router B and Router D have received update information containing the MED metric from Router E. • 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 example shows output from the **show ip route** command entered at Router E after this task has been configured and Router B and Router D have received update information containing the MED metric. Note that the gateway of last resort is set as 192.168.3.1, which is the route to Router B.

```
Router# show ip route
```

```
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route
Gateway of last resort is 192.168.3.1 to network 0.0.0.0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.2.2.0 is directly connected, Ethernet0/0
C    192.168.2.0/24 is directly connected, Serial3/0
C    192.168.3.0/24 is directly connected, Serial2/0
S*   0.0.0.0/0 [40/0] via 192.168.3.1
```

The following example shows output from the **show ip bgp** command entered at Router E after this task has been configured and Router B and Router D have received routing updates. The route map BLOCK has denied all routes coming in from autonomous system 45000 so the only network shown is the local network.

```
Router# show ip bgp

BGP table version is 2, local router ID is 10.2.2.99
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      0.0.0.0                  0         32768 i
```

The following example shows output from the **show ip bgp** command entered at Router B after this task has been configured at Router E and Router B has received routing updates. Note the metric of 50 for network 10.2.2.0.

```
Router# show ip bgp

BGP table version is 7, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 40000 i
*> 10.2.2.0/24      192.168.3.2             50             0 50000 i
*> 172.16.1.0/24    0.0.0.0                  0         32768 i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
```

The following example shows output from the **show ip bgp** command entered at Router D after this task has been configured at Router E and Router D has received routing updates. Note the metric of 100 for network 10.2.2.0.

```
Router# show ip bgp

BGP table version is 3, local router ID is 192.168.2.1
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      192.168.2.2            100             0 50000 i
*> 172.16.1.0/24    0.0.0.0                  0         32768 i
```

Configuring Multihoming to Receive the Full Internet Routing Table

Perform this task to configure your network to build neighbor relationships with other routers in other autonomous systems while filtering outbound routes. In this task the full Internet routing table will be received from the service providers in the neighboring autonomous systems but only locally originated routes will be advertised to the service providers. This task is configured at Router B in the figure above and uses an access list to permit only locally originated routes and a route map to ensure that only the locally originated routes are advertised outbound to other autonomous systems.

Note Be aware that receiving the full Internet routing table from two ISPs may use all the memory in smaller routers.

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*]
7. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
8. **exit**
9. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
10. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
11. **neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**}
12. **exit**
13. **exit**
14. **ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
15. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
16. **match as-path** *path-list-number*
17. **end**
18. **show ip bgp** [*network*] [*network-mask*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 5	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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*] **Example:** `Router(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.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Router(config-router-af)# neighbor 192.168.1.2 route-map localonly out`	Applies a route map to incoming or outgoing routes. • In this example, the route map named localonly is applied to outbound routes to Router A.
Step 8	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.

	Command or Action	Purpose
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(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 10	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(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 11	**neighbor** {*ip-address* | *peer-group-name*} **route-map** *map-name* {**in** | **out**} **Example:** `Router(config-router-af)# neighbor 192.168.3.2 route-map localonly out`	Applies a route map to incoming or outgoing routes. • In this example, the route map named localonly is applied to outbound routes to Router E.
Step 12	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 13	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 14	**ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression* **Example:** `Router(config)# ip as-path access-list 10 permit ^$`	Defines a BGP-related access list. • In this example, the access list number 10 is defined to permit only locally originated BGP routes.
Step 15	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*]	Configures a route map and enters route map configuration mode.

	Command or Action	Purpose
	Example: `Router(config)# route-map localonly permit 10`	• In this example, a route map named localonly is created.
Step 16	**match as-path** *path-list-number* **Example:** `Router(config-route-map)# match as-path 10`	Matches a BGP autonomous system path access list. • In this example, the BGP autonomous system path access list created in Step 12 is used for the match clause.
Step 17	**end** **Example:** `Router(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 18	**show ip bgp** [*network*] [*network-mask*] **Example:** `Router# show ip bgp`	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 example shows the BGP routing table for Router B in the figure above after this task has been configured. Note that the routing table contains the information about the networks in the autonomous systems 40000 and 50000.

```
BGP table version is 5, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.1.1.0/24      192.168.1.2              0             0 40000 i
*> 10.2.2.0/24      192.168.3.2              0             0 50000 i
*> 172.17.1.0/24    0.0.0.0                  0         32768 i
```

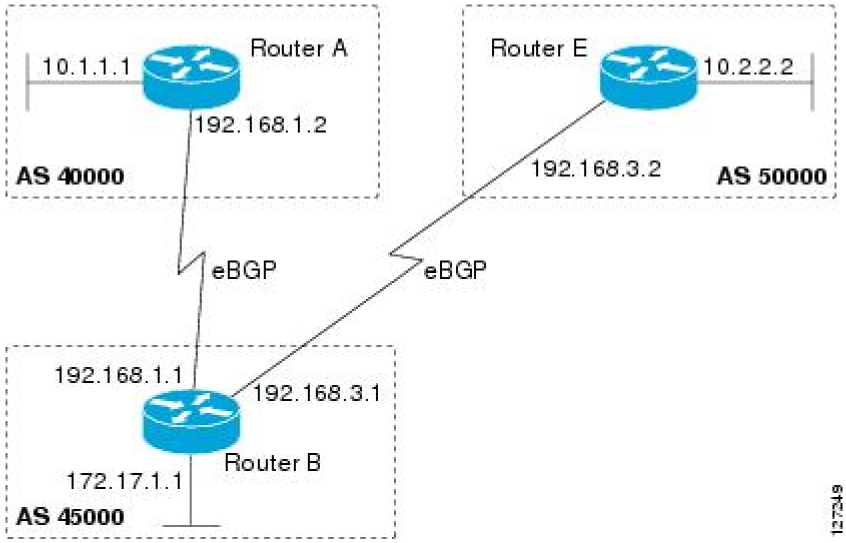
Configuring BGP Policies

The tasks in this section help you configure BGP policies that filter the traffic in your BGP network. The following optional tasks demonstrate some of the various methods by which traffic can be filtered in your BGP network:

Filtering BGP Prefixes with Prefix Lists

Perform this task to use prefix lists to filter BGP route information. The task is configured at Router B in the figure below where both Router A and Router E are set up as BGP peers. A prefix list is configured to permit only routes from the network 10.2.2.0/24 to be outbound. In effect, this will restrict the information that is received from Router E to be forwarded to Router A. Optional steps are included to display the prefix list information and to reset the hit count.

Figure 26: BGP Topology for Configuring BGP Policies Tasks



Note The **neighbor prefix-list** and the **neighbor distribute-list** commands are mutually exclusive for a BGP peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
5. Repeat Step 5 for all BGP peers.
6. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
7. **network** *network-number* [**mask** *network-mask*]
8. **aggregate-address** *address mask* [**as-set**]
9. **neighbor** *ip-address* **prefix-list** *list-name* {**in** | **out**}
10. **exit**
11. **exit**
12. **ip prefix-list** *list-name* [**seq** *seq-number*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] [**eq** *eq-value*]
13. **end**
14. **show ip prefix-list** [**detail** | **summary**] [*prefix-list-name* [**seq** *seq-number* | *network/length* [**longer** | **first-match**]]]
15. **clear ip prefix-list** {* | *ip-address* | *peer-group-name*} **out**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.2 remote-as 40000`	Adds the IP address of the neighbor in the specified autonomous system to the BGP neighbor table of the local router.
Step 5	Repeat Step 5 for all BGP peers.	--
Step 6	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Router(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 7	**network** *network-number* [**mask** *network-mask*] **Example:** `Router(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates.
Step 8	**aggregate-address** *address mask* [**as-set**] **Example:**	Creates an aggregate entry in a BGP routing table. • A specified route must exist in the BGP table.

	Command or Action	Purpose
	`Router(config-router-af)# aggregate-address 172.0.0.0 255.0.0.0`	• 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. **Note** Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 9	**neighbor** *ip-address* **prefix-list** *list-name* {**in** | **out**} **Example:** `Router(config-router-af)# neighbor 192.168.1.2 prefix-list super172 out`	Distributes BGP neighbor information as specified in a prefix list. • In this example, a prefix list called super172 is set for outgoing routes to Router A.
Step 10	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 11	**exit** **Example:** `Router(config-router) exit`	Exits router configuration mode and enters global configuration mode.
Step 12	**ip prefix-list** *list-name* [**seq** *seq-number*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] [**eq** *eq-value*] **Example:** `Router(config)# ip prefix-list super172 permit 172.0.0.0/8`	Defines a BGP-related prefix list and enters access list configuration mode. • In this example, the prefix list called super172 is defined to permit only route 172.0.0.0/8 to be forwarded. • All other routes will be denied because there is an implicit deny at the end of all prefix lists.
Step 13	**end** **Example:** `Router(config-access-list)# end`	Exits access list configuration mode and enters privileged EXEC mode.
Step 14	**show ip prefix-list** [**detail** | **summary**] [*prefix-list-name* [**seq** *seq-number* | *network/length* [**longer** | **first-match**]]] **Example:** `Router# show ip prefix-list detail super172`	Displays information about prefix lists. • In this example, details of the prefix list named super172 will be displayed, including the hit count. Hit count is the number of times the entry has matched a route.
Step 15	**clear ip prefix-list** {***** | *ip-address* | *peer-group-name*} **out**	Resets the hit count of the prefix list entries.

	Command or Action	Purpose
	Example: `Router# clear ip prefix-list super172 out`	• In this example, the hit count for the prefix list called super172 will be reset.

Examples

The following output from the **show ip prefix-list** command shows details of the prefix list named super172, including the hit count. The **clear ip prefix-list** command is entered to reset the hit count and the **show ip prefix-list** command is entered again to show the hit count reset to 0.

```
Router# show ip prefix-list detail super172

ip prefix-list super172:
   count: 1, range entries: 0, sequences: 5 - 5, refcount: 4
   seq 5 permit 172.0.0.0/8 (hit count: 1, refcount: 1)

Router# clear ip prefix-list super172

Router# show ip prefix-list detail super172

ip prefix-list super172:
   count: 1, range entries: 0, sequences: 5 - 5, refcount: 4
   seq 5 permit 172.0.0.0/8 (hit count: 0, refcount: 1)
```

Filtering BGP Prefixes with AS-Path Filters

Perform this task to filter BGP prefixes using AS-path filters with an access list based on the value of the AS-path attribute to filter route information. An AS-path access list is configured at Router B in the figure above. The first line of the access list denies all matches to AS-path 50000, and the second line allows all other paths. The router uses the **neighbor filter-list** command to specify the AS-path access list as an outbound filter. After the filter is enabled, traffic can be received from both Router A and Router C, but updates originating from autonomous system 50000 (Router C) are not forwarded by Router B to Router A. If any updates from Router C originated from another autonomous system, they would be forwarded because they would contain both autonomous system 50000 and another autonomous system number, and that would not match the AS-path access list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
4. Repeat Step 3 for all entries required in the AS-path access list.
5. **router bgp** *autonomous-system-number*
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. Repeat Step 6 for all BGP peers.
8. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
9. **neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out**}
10. **end**
11. **show ip bgp regexp** *as-regular-expression*

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 as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
	Example:
	`Device(config)# ip as-path access-list 100 deny ^50000$`
	Example:
	`Device(config)# ip as-path access-list 100 permit .*`
		Defines a BGP-related access list and enters access list configuration mode.
		• In the first example, access list number 100 is defined to deny any AS-path that starts and ends with 50000.
		• In the second example, all routes that do not match the criteria in the first example of the AS-path access list will be permitted. The period and asterisk symbols imply that all characters in the AS-path will match, so Router B will forward those updates to Router A.
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 4	Repeat Step 3 for all entries required in the AS-path access list.	—
Step 5	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
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 or peer group name of the neighbor in the specified autonomous system BGP neighbor table of the local router.
Step 7	Repeat Step 6 for all BGP peers.	—
Step 8	**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

	Command or Action	Purpose
		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 9	**neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number* {**in** | **out**} **Example:** `Device(config-router-af)# neighbor 192.168.1.2 filter-list 100 out`	Distributes BGP neighbor information as specified in a prefix list. • In this example, an access list number 100 is set for outgoing routes to Router A.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 11	**show ip bgp regexp** *as-regular-expression* **Example:** `Device# show ip bgp regexp ^50000$`	Displays routes that match the regular expression. • To verify the regular expression, you can use this command. • In this example, all paths that match the expression “starts and ends with 50000” will be displayed.

Examples

The following output from the **show ip bgp regexp** command shows the autonomous system paths that match the regular expression—start and end with AS-path 50000:

```
Device# show ip bgp regexp ^50000$

BGP table version is 9, 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           150 50000 i
```

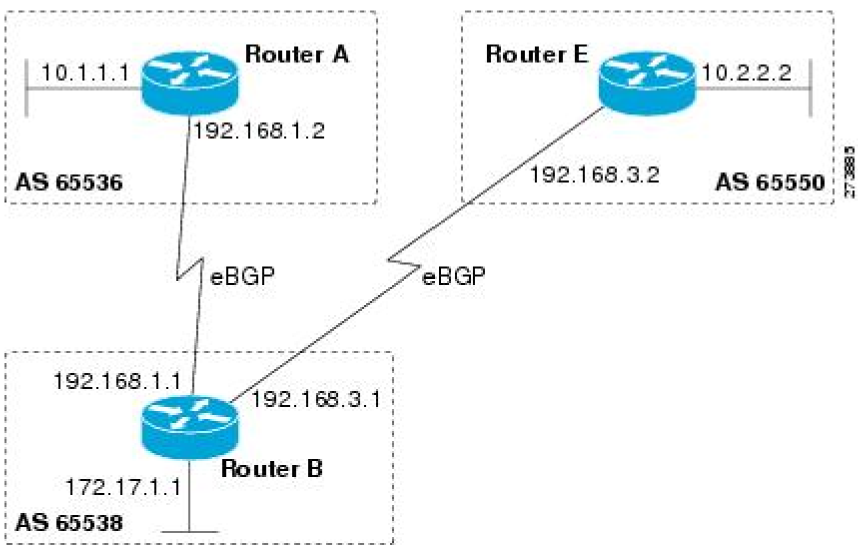
Filtering BGP Prefixes with AS-path Filters Using 4-Byte Autonomous System Numbers

In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and later releases, BGP support for 4-octet (4-byte) autonomous system numbers was introduced. The 4-byte autonomous system numbers in this task are formatted in the default asplain (decimal value) format, for example, Router B is in autonomous system number 65538 in the figure below. For more details about the introduction of 4-byte autonomous system numbers, see the "BGP Autonomous System Number Formats" section.

Perform this task to filter BGP prefixes with AS-path filters using 4-byte autonomous system numbers with an access list based on the value of the AS-path attribute to filter route information. An AS-path access list is configured at Router B in the figure below. The first line of the access list denies all matches to the AS-path 65550 and the second line allows all other paths. The router uses the **neighbor filter-list** command to specify the AS-path access list as an outbound filter. After the filtering is enabled, traffic can be received from both Router A and Router E but updates originating from autonomous system 65550 (Router E) are not forwarded by Router B to Router A. If any updates from Router E originated from another autonomous system, they would be forwarded because they would contain both autonomous system 65550 plus another autonomous system number, and that would not match the AS-path access list.

Note In Cisco IOS Releases 12.0(22)S, 12.2(15)T, 12.2(18)S, and later releases, the maximum number of autonomous system access lists that can be configured with the **ip as-path access-list** command is increased from 199 to 500.

Figure 27: BGP Topology for Filtering BGP Prefixes with AS-path Filters Using 4-Byte Autonomous System Numbers



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. Repeat Step 4 for all BGP peers.
6. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
7. **network** *network-number* [**mask** *network-mask*]
8. **neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number*{**in** | **out**}
9. **exit**
10. **exit**

11. **ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
12. Repeat Step 11 for all entries required in the AS-path access list.
13. **end**
14. **show ip bgp regexp** *as-regular-expression*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 65538`	Enters router configuration mode for the specified routing process.
Step 4	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.1.2 remote-as 65536`	Adds the IP address or peer group name of the neighbor in the specified autonomous system BGP neighbor table of the local router. • In this example, the IP address for the neighbor at Router A is added.
Step 5	Repeat Step 4 for all BGP peers.	--
Step 6	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(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.

	Command or Action	Purpose
Step 7	**network** *network-number* [**mask** *network-mask*] **Example:** `Router(config-router-af)# network 172.17.1.0 mask 255.255.255.0`	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. • For exterior protocols the **network** command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates. **Note** Only partial syntax is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **filter-list** *access-list-number*{**in** | **out**} **Example:** `Router(config-router-af)# neighbor 192.168.1.2 filter-list 99 out`	Distributes BGP neighbor information as specified in a prefix list. • In this example, an access list number 99 is set for outgoing routes to Router A.
Step 9	**exit** **Example:** `Router(config-router-af)# exit`	Exits address family configuration mode and returns to router configuration mode.
Step 10	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode and returns to global configuration mode.

Step 11	**ip as-path access-list** *access-list-number* {**deny** | **permit**} *as-regular-expression*
	Example:
	`Router(config)# ip as-path access-list 99 deny ^65550$`
	Example:
	`and`
	Example:
	`Router(config)# ip as-path access-list 99 permit .*`
		Defines a BGP-related access list and enters access list configuration mode.
		• In the first example, access list number 99 is defined to deny any AS-path that starts and ends with 65550.
		• In the second example, all routes that do not match the criteria in the first example of the AS-path access list will be permitted. The period and asterisk symbols imply that all characters in the AS-path will match, so Router B will forward those updates to Router A.
		Note Two examples are shown here because the task example requires both these statements to be configured.

Step 12	Repeat Step 11 for all entries required in the AS-path access list.	--
Step 13	**end** **Example:**	Exits access list configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Router(config-access-list)# end`	
Step 14	**show ip bgp regexp** *as-regular-expression* **Example:** `Router# show ip bgp regexp ^65550$`	Displays routes that match the regular expression. • To verify the regular expression, you can use this command. • In this example, all paths that match the expression "starts and ends with 65550" will be displayed.

Examples

The following output from the **show ip bgp regexp** command shows the autonomous system paths that match the regular expression--start and end with AS-path 65550:

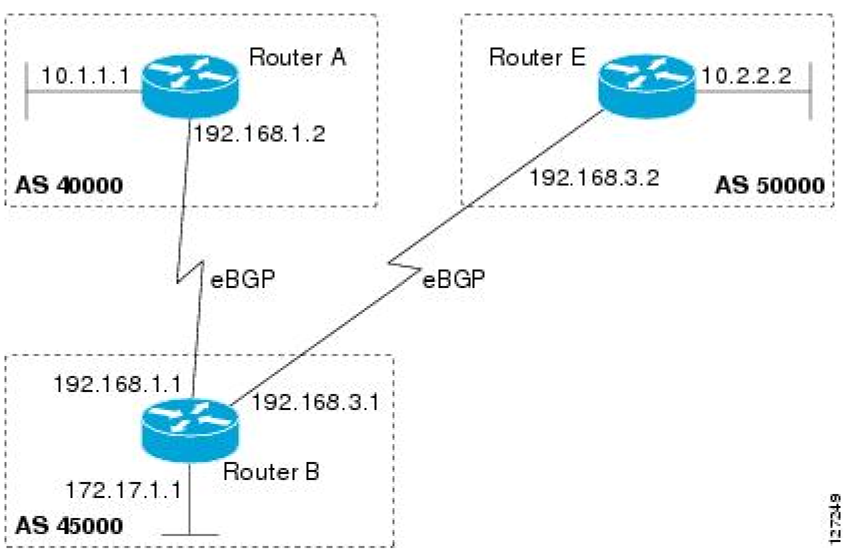
```
RouterB# show ip bgp regexp ^65550$
BGP table version is 4, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 10.2.2.0/24      192.168.3.2              0             0 65550 i
```

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 28: 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.
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:**	Adds the IP address or peer group name of the neighbor to the specified autonomous system BGP neighbor table of the local router.

	Command or Action	Purpose
	`Device(config-router)# neighbor 192.168.3.2 remote-as 50000`	
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*] **Example:** `Device(config)# route-map 2000 permit 10`	Creates a route map and enters route map configuration mode. • In this example, the route map called 2000 is defined.
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.

	Command or Action	Purpose
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*] [*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`
		Creates a community list for BGP and controls access to it.
		• 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.

	Command or Action	Purpose
		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
```

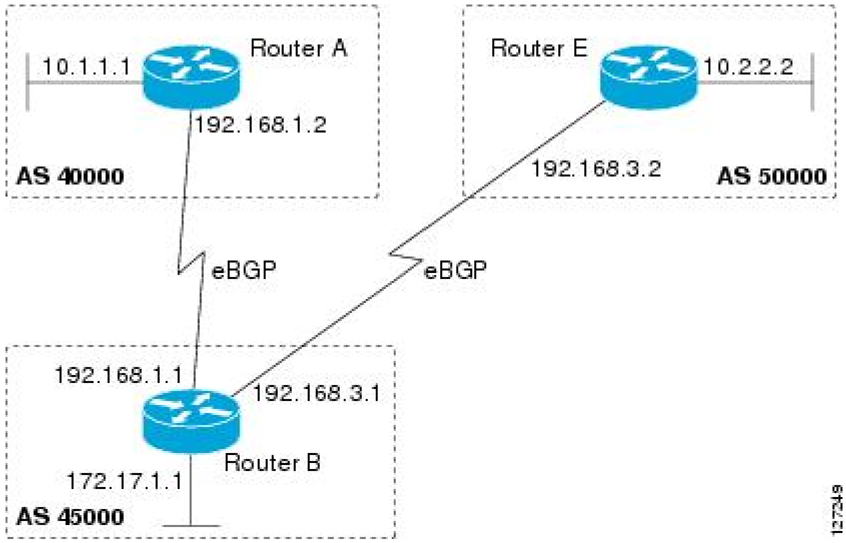
The following sample output verifies that community list 2 has been created and it effectively permits all routes by specifying the **internet** community:

```
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 29: 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*]}
	Example:
	`Device(config-extcomm-list)# 10 deny _50000_`
	Example:
	`Device(config-extcomm-list)# 20 deny ^50000 .*`
		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.
		• 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.

	Command or Action	Purpose
		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*] **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 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.

	Command or Action	Purpose
		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 .*
```

Filtering Traffic Using a BGP Route Map Policy List

Perform this task to create a BGP policy list and then reference it within a route map.

A policy list is like a route map that contains only match clauses. With policy lists there are no changes to match clause semantics and route map functions. The match clauses are configured in policy lists with permit and deny statements and the route map evaluates and processes each match clause to permit or deny routes based on the configuration. AND and OR semantics in the route map function the same way for policy lists as they do for match clauses.

Policy lists simplify the configuration of BGP routing policy in medium-size and large networks. The network operator can reference preconfigured policy lists with groups of match clauses in route maps and easily apply general changes to BGP routing policy. The network operator no longer needs to manually reconfigure each recurring group of match clauses that occur in multiple route map entries.

Perform this task to create a BGP policy list to filter traffic that matches the autonomous system path and MED of a router and then create a route map to reference the policy list.

Before you begin

BGP routing must be configured in your network and BGP neighbors must be established.

Note

- BGP route map policy lists do not support the configuration of IPv6 match clauses in policy lists.
- Policy lists are not supported in versions of Cisco IOS software prior to Cisco IOS Releases 12.0(22)S and 12.2(15)T. Reloading a router that is running an older version of Cisco IOS software may cause some routing policy configurations to be lost.
- Policy lists support only match clauses and do not support set clauses. However, policy lists can coexist, within the same route map entry, with match and set clauses that are configured separately from the policy lists.
- Policy lists are supported only by BGP. They are not supported by other IP routing protocols. This limitation does not interfere with normal operations of a route map, including redistribution, because policy list functions operate transparently within BGP and are not visible to other IP routing protocols.
- Policy lists support only match clauses and do not support set clauses. However, policy lists can coexist, within the same route map entry, with match and set clauses that are configured separately from the policy lists. The first route map example configures AND semantics, and the second route map configuration example configures semantics. Both examples in this section show sample route map configurations that reference policy lists and separate match and set clauses in the same configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip policy-list** *policy-list-name* {**permit** | **deny**}
4. **match as-path** *as-number*
5. **match metric** *metric*
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. **match policy-list** *policy-list-name*
10. **set community** *community-number* [**additive**] [*well-known-community*] | **none**}
11. **set local-preference** *preference-value*
12. **end**
13. **show ip policy-list** [*policy-list-name*]
14. **show route-map** [*route-map-name*]

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**ip policy-list** *policy-list-name* {**permit** | **deny**} **Example:** `Router(config)# ip policy-list POLICY-LIST-NAME-1 permit`	Enters policy list configuration mode and creates a BGP policy list that will permit routes that are allowed by the match clauses that follow.
Step 4	**match as-path** *as-number* **Example:** `Router(config-policy-list)# match as-path 500`	Creates a match clause to permit routes from the specified autonomous system path.
Step 5	**match metric** *metric* **Example:** `Router(config-policy-list)# match metric 10`	Creates a match clause to permit routes with the specified metric.
Step 6	**exit** **Example:** `Router(config-policy-list)# exit`	Exits policy list configuration mode and enters global configuration mode.
Step 7	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map MAP-NAME-1 permit 10`	Creates a route map and enters route map configuration mode.
Step 8	**match ip address** {*access-list-number* | *access-list-name*} [... *access-list-number* | ... *access-list-name*] **Example:** `Router(config-route-map)# match ip address 1`	Creates a match clause to permit routes that match the specified *access-list-number*or *access-list-name*argument.
Step 9	**match policy-list** *policy-list-name* **Example:** `Router(config-route-map)# match policy-list POLICY-LIST-NAME-1`	Creates a clause that will match the specified policy list. • All match clauses within the policy list will be evaluated and processed. Multiple policy lists can referenced with this command. • This command also supports AND or OR semantics like a standard match clause.
Step 10	**set community** *community-number* [**additive**] [*well-known-community*] | **none**} **Example:** `Router(config-route-map)# set community 10:1`	Creates a clause to set or remove the specified community.

	Command or Action	Purpose
Step 11	**set local-preference** *preference-value* **Example:** `Router(config-route-map)# set local-preference 140`	Creates a clause to set the specified local preference value.
Step 12	**end** **Example:** `Router(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 13	**show ip policy-list** [*policy-list-name*] **Example:** `Router# show ip policy-list POLICY-LIST-NAME-1`	Display information about configured policy lists and policy list entries.
Step 14	**show route-map** [*route-map-name*] **Example:** `Router# show route-map`	Displays locally configured route maps and route map entries.

Examples

The following sample output verifies that a policy list has been created, with the output displaying the policy list name and configured match clauses:

```
Router# show ip policy-list
 POLICY-LIST-NAME-1

policy-list POLICY-LIST-NAME-1 permit
  Match clauses:
    metric 20
    as-path (as-path filter): 1
```

Note A policy list name can be specified when the **show ip policy-list** command is entered. This option can be useful for filtering the output of this command and verifying a single policy list.

The following sample output from the **show route-map** command verifies that a route map has been created and a policy list is referenced. The output of this command displays the route map name and policy lists that are referenced by the configured route maps.

```
Router# show route-map

route-map ROUTE-MAP-NAME-1, deny, sequence 10
  Match clauses:
  Set clauses:
  Policy routing matches: 0 packets, 0 bytes
route-map ROUTE-MAP-NAME-1, permit, sequence 10
```

```
Match clauses:
  IP Policy lists:
      POLICY-LIST-NAME-1
Set clauses:
Policy routing matches: 0 packets, 0 bytes
```

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

	Command or Action	Purpose
	`Device(config-router)# neighbor 10.0.0.1 remote-as 50000`	
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.
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

	Command or Action	Purpose
		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 Connecting to a Service Provider Using External BGP

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
```

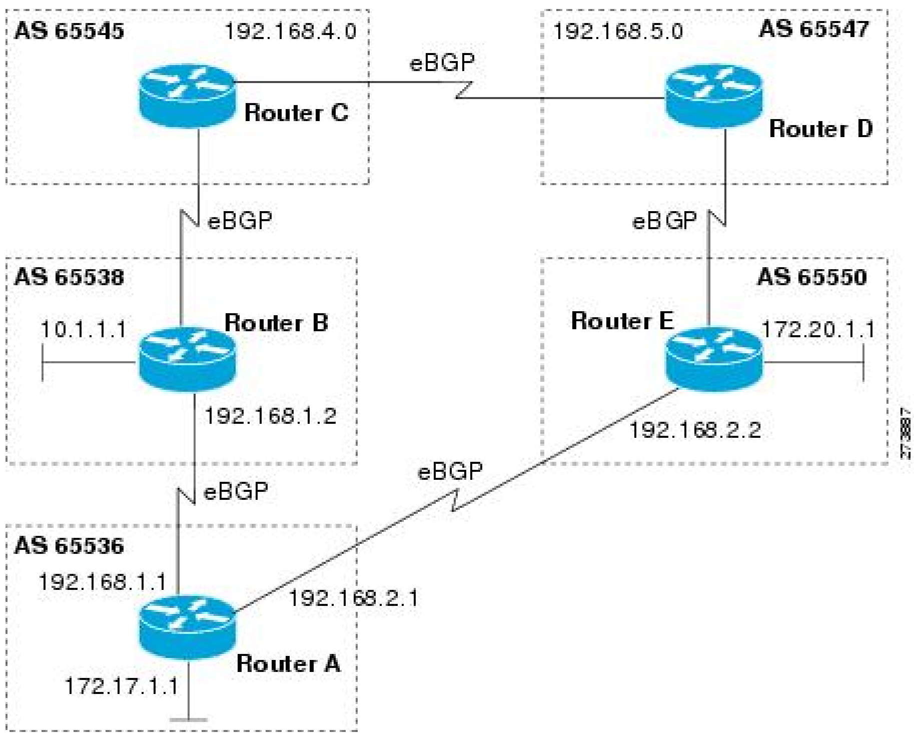
Example: Influencing Inbound Path Selection by Modifying the AS-path Attribute Using 4-Byte AS Numbers

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 Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, 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. For more details about the introduction of 4-byte autonomous system numbers, see the “BGP Autonomous System Number Formats” section.

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 30: 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: 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
```

Example: Filtering BGP Prefixes with Prefix Lists

This section contains the following examples:

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
```

Example: Filtering BGP Prefixes Using a Group of Prefixes

The following example shows how to configure BGP to permit routes with a prefix length up to 24 in network 192/8:

```
ip prefix-list abc permit 192.0.0.0/8 le 24
```

The following example shows how to configure BGP to deny routes with a prefix length greater than 25 in 192/8:

```
ip prefix-list abc deny 192.0.0.0/8 ge 25
```

The following example shows how to configure BGP to permit routes with a prefix length greater than 8 and less than 24 in all address space:

```
ip prefix-list abc permit 0.0.0.0/0 ge 8 le 24
```

The following example shows how to configure BGP to deny routes with a prefix length greater than 25 in all address space:

```
ip prefix-list abc deny 0.0.0.0/0 ge 25
```

The following example shows how to configure BGP to deny all routes in network 10/8, because any route in the Class A network 10.0.0.0/8 is denied if its mask is less than or equal to 32 bits:

```
ip prefix-list abc deny 10.0.0.0/8 le 32
```

The following example shows how to configure BGP to deny routes with a mask greater than 25 in 192.168.1.0/24:

```
ip prefix-list abc deny 192.168.1.0/24 ge 25
```

The following example shows how to configure BGP to permit all routes:

```
ip prefix-list abc permit 0.0.0.0/0 le 32
```

Example: Adding or Deleting Prefix List Entries

You can add or delete individual entries in a prefix list if a prefix list has the following initial configuration:

```
ip prefix-list abc deny 0.0.0.0/0 le 7
ip prefix-list abc deny 0.0.0.0/0 ge 25
ip prefix-list abc permit 192.168.0.0/15
```

The following example shows how to delete an entry from the prefix list so that 192.168.0.0 is not permitted, and add a new entry that permits 10.0.0.0/8:

```
no ip prefix-list abc permit 192.168.0.0/15
ip prefix-list abc permit 10.0.0.0/8
```

The new configuration is as follows:

```
ip prefix-list abc deny 0.0.0.0/0 le 7
ip prefix-list abc deny 0.0.0.0/0 ge 25
ip prefix-list abc permit 10.0.0.0/8
```

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 .*
```

Example: Filtering Traffic Using AS-Path Filters

The following example shows BGP path filtering by neighbor. Only the routes that pass autonomous system path access list 2 will be sent to 192.168.12.10. Similarly, only routes passing access list 3 will be accepted from 192.168.12.10.

```
router bgp 200
 neighbor 192.168.12.10 remote-as 100
 neighbor 192.168.12.10 filter-list 1 out
 neighbor 192.168.12.10 filter-list 2 in
 exit
ip as-path access-list 1 permit _109_
ip as-path access-list 2 permit _200$
ip as-path access-list 2 permit ^100$
ip as-path access-list 3 deny _690$
ip as-path access-list 3 permit .*
```

Example: Filtering Traffic with AS-path Filters Using 4-Byte Autonomous System Numbers

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example is available in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)XNE, 12.2(33)SXI1, and later releases and shows BGP path filtering by neighbor using 4-byte autonomous system numbers in asplain format. Only the routes that pass autonomous system path access list 2 will be sent to 192.168.3.2.

```
ip as-path access-list 2 permit ^65536$
router bgp 65538
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 65550
  neighbor 192.168.3.2 activate
  neighbor 192.168.3.2 filter-list 2 in
  end
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T

The following example available in Cisco IOS Release 12.0(32)S12, 12.4(24)T, and later releases shows BGP path filtering by neighbor using 4-byte autonomous system numbers in asdot format. Only the routes that pass autonomous system path access list 2 will be sent to 192.168.3.2.

Note 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, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

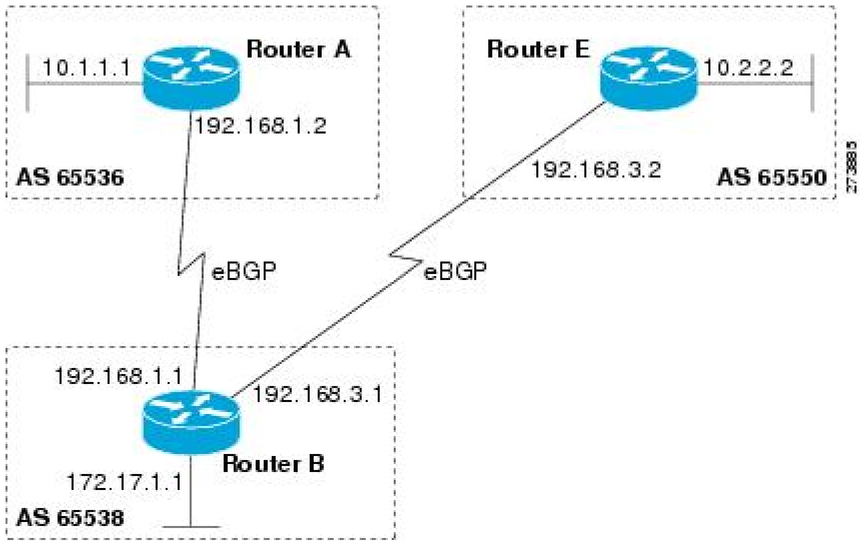
```
ip as-path access-list 2 permit ^1\.0$
router bgp 1.2
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 1.14
  neighbor 192.168.3.2 filter-list 2 in
  end
```

Example: Filtering Traffic Using Extended Community Lists with 4-Byte Autonomous System Numbers

Asplain Default Format in Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SRE, 12.2(33)SXI1, and Later Releases

The following example shows how to filter traffic by creating an extended BGP community list to control outbound routes. 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, extended BGP communities support 4-byte autonomous system numbers in the regular expressions in asplain by default. Extended community attributes are used to configure, filter, and identify routes for VRF instances and MPLS VPNs. The **ip extcommunity-list**command is used to configure named or numbered extended community lists. All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers.

Figure 31: BGP Topology for Filtering Traffic Using Extended Community Lists with 4-Byte Autonomous System Numbers in Asplain Format



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 and not in global configuration mode.

In this exam the figure above is configured with an extended named community list to specify that the BGP peer at 192.1681.2 is not sent advertisements about any path through or from the 4-byte autonomous system 65550. The IP extended community-list configuration mode is used, and the ability to resequence entries is shown.

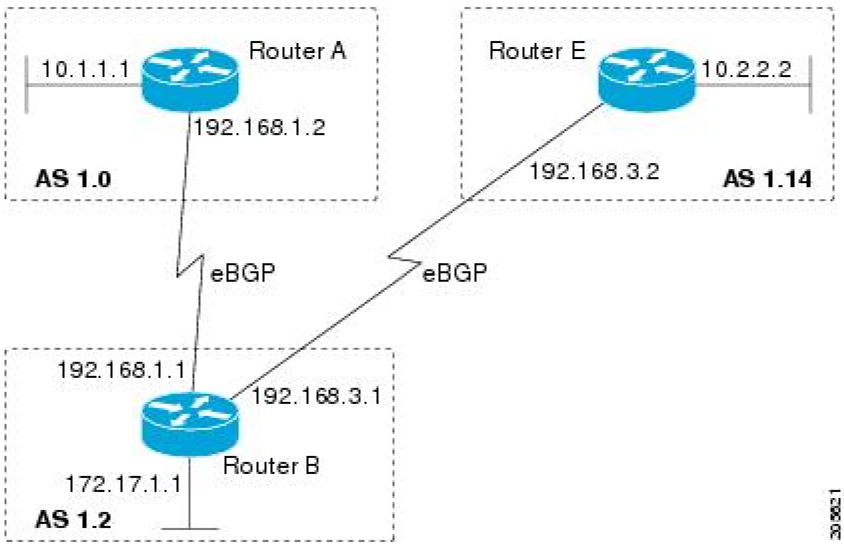
```
ip extcommunity-list expanded DENY65550
 10 deny _65550_
 20 deny ^65550 .*
 resequence 50 100
 exit
router bgp 65538
 network 172.17.1.0 mask 255.255.255.0
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 65550
  neighbor 192.168.1.2 remote-as 65536
  neighbor 192.168.3.2 activate
  neighbor 192.168.1.2 activate
  end
show ip extcommunity-list DENY65550
```

Asdot Default Format in Cisco IOS Release 12.0(32)S12, and 12.4(24)T

The following example shows how to filter traffic by creating an extended BGP community list to control outbound routes. In Cisco IOS Release 12.0(32)S12, 12.4(24)T, and later releases, extended BGP communities support 4-byte autonomous system numbers in the regular expressions in asdot format only. Extended community attributes are used to configure, filter, and identify routes for VRF instances and MPLS VPNs. The **ip extcommunity-list**command is used to configure named or numbered extended community lists. All of the standard rules of access lists apply to the configuration of extended community lists. Regular expressions are supported by the expanded range of extended community list numbers.

Note In Cisco IOS Release 12.0(32)SY8, 12.0(33)S3, 12.2(33)SXI1, and later releases, this example works if you have configured asdot as the default display format using the **bgp asnotation dot** command.

Figure 32: BGP Topology for Filtering Traffic Using Extended Community Lists with 4-Byte Autonomous System Numbers in Asdot Format



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 and not in global configuration mode.

In this exam the figure above is configured with an extended named community list to specify that the BGP peer at 192.1681.2 is not sent advertisements about any path through or from the 4-byte autonomous system 65550. The IP extended community-list configuration mode is used, and the ability to resequence entries is shown.

```
ip extcommunity-list expanded DENY114
 10 deny _1\.14_
 20 deny ^1\.14 .*
 resequence 50 100
 exit
router bgp 1.2
 network 172.17.1.0 mask 255.255.255.0
 address-family ipv4 unicast
  neighbor 192.168.3.2 remote-as 1.14
  neighbor 192.168.1.2 remote-as 1.0
  neighbor 192.168.3.2 activate
  neighbor 192.168.1.2 activate
```

```
  end
show ip extcommunity-list DENY114
```

Example: Filtering Traffic Using a BGP Route Map

The following example shows how to use an address family to configure BGP so that any unicast and multicast routes from neighbor 10.1.1.1 are accepted if they match access list 1:

```
route-map filter-some-multicast
 match ip address 1
 exit
router bgp 65538
 neighbor 10.1.1.1 remote-as 65537
 address-family ipv4 unicast
  neighbor 10.1.1.1 activate
  neighbor 10.1.1.1 route-map filter-some-multicast in
  exit
 exit
router bgp 65538
 neighbor 10.1.1.1 remote-as 65537
 address-family ipv4 multicast
  neighbor 10.1.1.1 activate
  neighbor 10.1.1.1 route-map filter-some-multicast in
  end
```

Where to Go Next

- To configure advanced BGP feature tasks, proceed to the “Configuring Advanced BGP Features” module.
- To configure BGP neighbor session options, proceed to the “Configuring BGP Neighbor Session Options” module.
- To configure internal BGP tasks, proceed to the “Configuring Internal BGP Features” module.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands: complete command syntax, command mode, defaults, command history, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
BGP overview	“Cisco BGP Overview” module
Configuring basic BGP tasks	“Configuring a Basic BGP Network” module
BGP fundamentals and description	*Large-Scale IP Network Solutions*, Khalid Raza and Mark Turner, Cisco Press, 2000

Related Topic	Document Title
Implementing and controlling BGP in scalable networks	*Building Scalable Cisco Networks*, Catherine Paquet and Diane Teare, Cisco Press, 2001
Interdomain routing basics	*Internet Routing Architectures*, Bassam Halabi, Cisco Press, 1997

Standards

Standard	Title
MDT SAFI	MDT SAFI

MIBs

MIB	MIBs Link
CISCO-BGP4-MIB	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 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 of 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)*
RFC 4893	*BGP Support for Four-Octet AS Number Space*
RFC 5396	*Textual Representation of Autonomous system (AS) Numbers*
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*

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 Connecting to a Service Provider Using External BGP

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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 23: Feature Information for Connecting to a Service Provider Using External BGP

Feature Name	Releases	Feature Configuration Information
BGP Increased Support of Numbered AS-Path Access Lists to 500	12.0(22)S 12.2(15)T 12.2(18)S 12.2(18)SXD 12.2(27)SBC 15.0(1)S	The BGP Increased Support of Numbered AS-Path Access Lists to 500 feature increases the maximum number of autonomous systems access lists that can be configured using the **ip as-path access-list** command from 199 to 500.
BGP Named Community Lists	12.2(8)T 12.2(14)S 15.0(1)S	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.

Feature Name	Releases	Feature Configuration Information
BGP Route-Map Policy List Support	12.0(22)S 12.2(15)T 12.2(18)S 12.2(18)SXD 12.2(27)SBC 15.0(1)S	The BGP Route-Map Policy List Support feature introduces new functionality to BGP route maps. This feature adds the capability for a network operator to group route map match clauses into named lists called policy lists. A policy list functions like a macro. When a policy list is referenced in a route map, all of the match clauses are evaluated and processed as if they had been configured directly in the route map. This enhancement simplifies the configuration of BGP routing policy in medium-size and large networks because a network operator can preconfigure policy lists with groups of match clauses and then reference these policy lists within different route maps. The network operator no longer needs to manually reconfigure each recurring group of match clauses that occur in multiple route map entries.
BGP Support for Named Extended Community Lists	12.2(25)S 12.2(27)SBC 12.2(33)SRA 12.2(33)SXH 12.3(11)T 15.0(1)S	The BGP Support for Named Extended Community Lists feature introduces the ability to configure extended community lists using names in addition to the existing numbered format.
BGP Support for Sequenced Entries in Extended Community Lists	12.2(25)S 12.2(27)SBC 12.2(33)SRA 12.2(33)SXH 12.3(11)T 15.0(1)S	The BGP Support for Sequenced Entries in Extended Community Lists feature introduces automatic sequencing of individual entries in BGP extended community lists. This feature also introduces the ability to remove or resequence extended community list entries without deleting the entire existing extended community list.
BGP 4 Prefix Filter and Inbound Route Maps	Cisco IOS XE 3.1.0SG	

CHAPTER 11

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.
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.

	Command or Action	Purpose
		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**} **Example:** `Device(config-router-af)# neighbor 10.1.1.1 prefix-list FILTER in`	Applies an inbound prefix-list filter to prevent distribution of BGP neighbor information. • 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:**	Clears BGP outbound route filters and initiates an inbound soft reset.

	Command or Action	Purpose
	`Device# clear ip bgp 10.1.1.1 in prefix-filter`	• 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 24: Feature Information for BGP Prefix-Based Outbound Route Filtering

Feature Name	Releases	Feature Information
BGP Prefix-Based Outbound Route Filtering	12.2(4)T	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.

CHAPTER 12

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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
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*] **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*.

	Command or Action	Purpose
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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 25: Feature Information for BGP Route Map Continue

Feature Name	Releases	Feature Information
BGP Route Map Continue	12.3(2)T	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.

CHAPTER 13

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:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
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 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.

	Command or Action	Purpose
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.
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.

	Command or Action	Purpose
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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 26: Feature Information for BGP Route-Map Continue Support for Outbound Policy

Feature Name	Releases	Feature Information
BGP Route-Map Continue Support for Outbound Policy	12.4(4)T	The BGP Route-Map Continue Support for an Outbound Policy feature introduces support for continue clauses to be applied to outbound route maps.

CHAPTER 14

Removing Private AS Numbers from the AS Path in BGP

Private autonomous system numbers (ASNs) are used by ISPs and customer networks to conserve globally unique AS numbers. Private AS numbers cannot be used to access the global Internet because they are not unique. AS numbers appear in eBGP AS paths in routing updates. Removing private ASNs from the AS path is necessary if you have been using private ASNs and you want to access the global Internet.

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.

Restrictions on Removing and Replacing Private ASNs from the AS Path

- The feature applies to eBGP neighbors only.
- The feature applies to routers in a public AS only. The workaround to this restriction would be to apply the **neighbor local-as** command on a per-neighbor basis, with the local AS number being a public AS number.

Information About Removing and Replacing Private ASNs from the AS Path

Public and Private AS Numbers

Public AS numbers are assigned by InterNIC and are globally unique. They range from 1 to 64511. Private AS numbers are used to conserve globally unique AS numbers, and they range from 64512 to 65535. Private AS numbers cannot be leaked to a global BGP routing table because they are not unique, and BGP best path calculations require unique AS numbers. Therefore, it might be necessary to remove private AS numbers from an AS path before the routes are propagated to a BGP peer.

Benefit of Removing and Replacing Private ASNs from the AS Path

External BGP requires that globally unique AS numbers be used when routing to the global Internet. Using private AS numbers (which are not unique) would prevent access to the global Internet. This feature allows routers that belong to a private AS to access the global Internet. A network administrator configures the routers to remove private AS numbers from the AS path contained in outgoing update messages and optionally, to replace those numbers with the ASN of the local router, so that the AS Path length remains unchanged.

Former Restrictions to Removing Private ASNs from the AS Path

The ability to remove private AS numbers from the AS path has been available for a long time. Prior to Cisco IOS Release 15.1(2)T, this feature had the following restrictions:

- If the AS path included both private and public AS numbers, using the **neighbor remove-private-as** command would not remove the private AS numbers.
- If the AS path contained confederation segments, using the **neighbor remove-private-as** command would remove private AS numbers only if the private AS numbers followed the confederation portion of the autonomous path.
- If the AS path contained the AS number of the eBGP neighbor, the private AS numbers would not be removed.

Enhancements to Removing Private ASNs from the AS Path

The ability to remove and replace private AS numbers from the AS path is enhanced in the following ways:

- The **neighbor remove-private-as** command will remove private AS numbers from the AS path even if the path contains both public and private ASNs.
- The **neighbor remove-private-as** command will remove private AS numbers even if the AS path contains only private AS numbers. There is no likelihood of a 0-length AS path because this command can be applied to eBGP peers only, in which case the AS number of the local router is appended to the AS path.
- The **neighbor remove-private-as** command will remove private AS numbers even if the private ASNs appear before the confederation segments in the AS path.

- The **replace-as** keyword is available to replace the private AS numbers being removed from the path with the local AS number, thereby retaining the same AS path length.
- The feature can be applied to neighbors per address family (address family configuration mode). Therefore, you can apply the feature for a neighbor in one address family and not on another, affecting update messages on the outbound side for only the address family for which the feature is configured.
- The feature can be applied in peer group template mode.
- When the feature is configured, output from the **show ip bgp update-group** and **show ip bgp neighbor** commands indicates that private AS numbers were removed or replaced.

How to Remove and Replace Private ASNs from the AS Path

Removing and Replacing Private ASNs from the AS Path (Cisco IOS Release 15.1(2)T and Later)

To remove private AS numbers from the AS path on the outbound side of an eBGP neighbor, perform the following task. To also replace private AS numbers with the local router's AS number, include the **all replace-as** keywords in Step 17.

The examples in this task reflect the configuration for Router 2 in the scenario in the figure below.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask*
5. **exit**
6. **interface** *type number*
7. **ip address** *ip-address mask*
8. **exit**
9. **interface** *type number*
10. **ip address** *ip-address mask*
11. **exit**
12. **router bgp** *autonomous-system-number*
13. **network** *network-number*
14. **network** *network-number*
15. **neighbor** {*ip-address* | *ipv6-address*[**%**]| *peer-group-name*} **remote-as** *autonomous-system-number*
16. **neighbor** {*ip-address* | *ipv6-address*[**%**]| *peer-group-name*} **remote-as** *autonomous-system-number*
17. **neighbor** {*ip-address* | *peer-group-name*} **remove-private-as** [**all** [**replace-as**]]
18. **end**
19. **show ip bgp update-group**
20. **show ip bgp neighbors**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface gigabitethernet 0/0`	Configures an interface.
Step 4	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 172.30.1.1 255.255.0.0`	Sets a primary or secondary IP address for an interface.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Returns to the next highest configuration mode.
Step 6	**interface** *type number* **Example:** `Router(config)# interface serial 0/0`	Configures an interface.
Step 7	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 172.16.0.2 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 8	**exit** **Example:** `Router(config-if)# exit`	Returns to the next highest configuration mode.
Step 9	**interface** *type number* **Example:** `Router(config)# interface serial 1/0`	Configures an interface.

	Command or Action	Purpose
Step 10	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 192.168.0.1 255.255.255.0`	Sets a primary or secondary IP address for an interface.
Step 11	**exit** **Example:** `Router(config-if)# exit`	Returns to the next highest configuration mode.
Step 12	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Specifies a BGP instance.
Step 13	**network** *network-number* **Example:** `Router(config-router)# network 172.30.0.0`	Specifies a network to be advertised by BGP.
Step 14	**network** *network-number* **Example:** `Router(config-router)# network 192.168.0.0`	Specifies a network to be advertised by BGP.
Step 15	**neighbor** {*ip-address* | *ipv6-address*[**%**]| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 172.16.0.1 remote-as 65000`	Adds an entry to the routing table. • This example configures Router 3 as an eBGP neighbor in private AS 65000.
Step 16	**neighbor** {*ip-address* | *ipv6-address*[**%**]| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.0.2 remote-as 1`	Adds an entry to the routing table. • This example configures Router 1 as an eBGP neighbor in public AS 1.
Step 17	**neighbor** {*ip-address* | *peer-group-name*} **remove-private-as** [**all** [**replace-as**]] **Example:** `Router(config-router)# neighbor 192.168.0.2 remove-private-as all replace-as`	Removes private AS numbers from the AS Path in outgoing updates. • This example removes the private AS numbers from the AS path in outgoing eBGP updates and replaces them with 5, which is the public AS number of the local router.

	Command or Action	Purpose
Step 18	**end** **Example:** `Router(config-router)# end`	Ends the current configuration mode and returns to privileged EXEC mode.
Step 19	**show ip bgp update-group** **Example:** `Router# show ip bgp update-group`	(Optional) Displays information about BGP update groups.
Step 20	**show ip bgp neighbors** **Example:** `Router# show ip bgp neighbors`	(Optional) Displays information about BGP neighbors.

Configuration Examples for Removing and Replacing Private ASNs from the AS Path

Example: Removing Private ASNs (Cisco IOS Release 15.1(2)T)

In the example below, Router A has the **neighbor remove-private-as** command configured, which removes private AS numbers in updates sent to the neighbor at 172.30.0.7. The subsequent **show** command asks for information about the route to host 1.1.1.1. The output includes private AS numbers 65200, 65201, 65201 in the AS path of 1001 65200 65201 65201 1002 1003 1003.

To prove that the private AS numbers were removed from the AS path, the **show** command on Router B also asks for information about the route to host 1.1.1.1. The output indicates a shorter AS path of 100 1001 1002 1003 1003, which excludes private AS numbers 65200, 65201, and 65201. The 100 prepended in the path is Router B's own AS number.

Router A

```
router bgp 100
 bgp log-neighbor-changes
 neighbor 19.0.101.1 remote-as 1001
 neighbor 172.30.0.7 remote-as 200
 neighbor 172.30.0.7 remove-private-as all
 no auto-summary

RouterA# show ip bgp 1.1.1.1
BGP routing table entry for 1.1.1.1/32, version 2
Paths: (1 available, best #1, table default)
  Advertised to update-groups:
     1          2
  1001 65200 65201 65201 1002 1003 1003
    19.0.101.1 from 19.0.101.1 (19.0.101.1)
      Origin IGP, localpref 100, valid, external, best RouterA#
```

Router B (All Private ASNs Have Been Removed)

```
RouterB# show ip bgp 1.1.1.1

BGP routing table entry for 1.1.1.1/32, version 3
Paths: (1 available, best #1, table default)
  Not advertised to any peer
  100 1001 1002 1003 1003
    172.30.0.6 from 172.30.0.6 (19.1.0.1)
      Origin IGP, localpref 100, valid, external, best RouterB#
```

Example: Removing and Replacing Private ASNs (Cisco IOS Release 15.1(2)T)

In the following example, when Router A sends prefixes to the peer 172.30.0.7, all private ASNs in the AS path are replaced with the router's own ASN, which is 100.

Router A

```
router bgp 100
 bgp log-neighbor-changes
 neighbor 172.16.101.1 remote-as 1001
 neighbor 172.16.101.1 update-source Loopback0
 neighbor 172.30.0.7 remote-as 200
 neighbor 172.30.0.7 remove-private-as all replace-as
 no auto-summary
```

Router A receives 1.1.1.1 from peer 172.16.101.1 which has some private ASNs (65200, 65201, and 65201) in the AS path list, as shown in the following output:

```
RouterA# show ip bgp 1.1.1.1

BGP routing table entry for 1.1.1.1/32, version 2
Paths: (1 available, best #1, table default)
   Advertised to update-groups:
      1          2
   1001 65200 65201 65201 1002 1003 1003
     172.16.101.1 from 172.16.101.1 (172.16.101.1)
       Origin IGP, localpref 100, valid, external, best RouterA#
```

Because Router A is configured with **neighbor 172.30.0.7 remove-private-as all replace-as**, Router A sends prefix 1.1.1.1 with all private ASNs replaced with 100:

Router B

```
RouterB# show ip bgp 1.1.1.1

BGP routing table entry for 1.1.1.1/32, version 3
Paths: (1 available, best #1, table default)
   Not advertised to any peer
   100 1001 100 100 100 1002 1003 1003
     172.30.0.6 from 172.30.0.6 (192.168.1.2)
       Origin IGP, localpref 100, valid, external, best RouterB#
```

Router B

```
router bgp 200
```

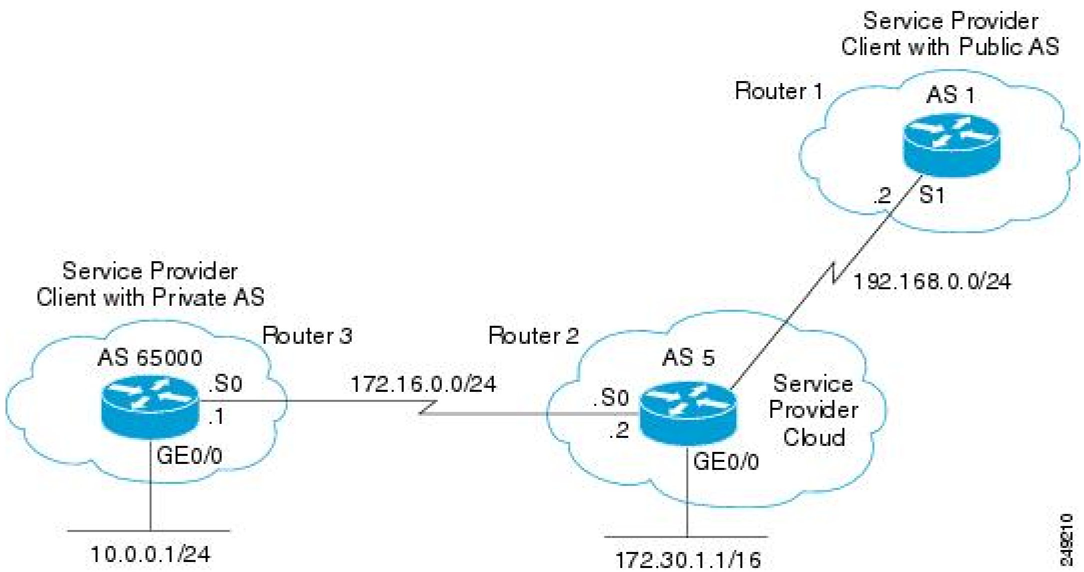
```
 bgp log-neighbor-changes
 neighbor 172.30.0.6 remote-as 100
 no auto-summary
```

Example: Removing Private ASNs (Cisco IOS Release 12.2)

In this example, Router 3 uses private ASN 65000. Router 1 and Router 2 use public ASNs AS 1 and AS 5 respectively.

The figure below illustrates Router 2 belonging to a service provider, with Router 1 and Router 3 as its clients.

Figure 33: Removing Private AS Numbers



In this example, Router 2, belonging to the Service Provider, removes private AS numbers as follows.

1. Router 3 advertises the network 10.0.0.0/24 with the AS path attribute 65000 to Router 2.
2. Router 2 receives the update from Router 3 and makes an entry for the network 10.0.0.0/24 in its routing table with the next hop as 172.16.0.1 (serial interface S0 on Router 3).
3. Router 2 (service provider device), when configured with the **neighbor 192.168.0.2 remove-private-as** command, strips off the private AS number and constructs a new update packet with its own AS number as the AS path attribute for the 10.0.0.0/24 network and sends the packet to Router 1.
4. Router 1 receives the eBGP update for network 10.0.0.0/24 and makes an entry in its routing table with the next hop as 192.168.0.1 (serial interface S1 on Router 2). The AS path attribute for this network as seen on Router 1 is AS 5 (Router 2). Thus, the private AS numbers are prevented from entering the BGP tables of the Internet.

The configurations of Router 3, Router 2, and Router 1 follow.

Router 3

```
interface gigabitethernet 0/0
 ip address 10.0.0.1 255.255.255.0
!
```

```
interface Serial 0
 ip address 172.16.0.1 255.255.255.0
!
router bgp 65000
 network 10.0.0.0 mask 255.255.255.0
 neighbor 172.16.0.2 remote-as 5
!---Configures Router 2 as an eBGP neighbor in public AS 5.
!
end
```

Router 2

```
interface gigabitethernet 0/0
 ip address 172.30.1.1 255.255.0.0
!
interface Serial 0
 ip address 172.16.0.2 255.255.255.0
!
interface Serial 1
 ip address 192.168.0.1 255.255.255.0
!
router bgp 5
 network 172.30.0.0
 network 192.168.0.0
 neighbor 172.16.0.1 remote-as 65000
!---Configures Router 3 as an eBGP neighbor in private AS 65000.
 neighbor 192.168.0.2 remote-as 1
!---Configures Router 1 as an eBGP neighbor in public AS 1.
 neighbor 192.168.0.2 remove-private-as
!---Removes the private AS numbers from outgoing eBGP updates.
!
end
```

Router 1

```
version 12.2
!
!
interface Serial 0
 ip address 192.168.0.2 255.255.255.0
!
router bgp 1
 neighbor 192.168.0.1 remote-as 5
!---Configures Router 2 as an eBGP neighbor in public AS 5.
!
end
```

Additional References

Related Documents

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

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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 Removing and Replacing Private ASNs from the AS Path

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 27: Feature Information for BGP—Remove/Replace Private AS

Feature Name	Releases	Feature Information
BGP--Remove/Replace Private AS	15.1(2)T 15.0(1)S 15.1(1)SY	Private autonomous system (AS) numbers are used by ISPs and customer networks to conserve globally unique AS numbers. Private AS numbers cannot be used to access the global Internet because they are not unique. AS numbers appear in eBGP AS paths in routing tables. Removing private AS numbers from the AS path is necessary if you have been using private AS numbers and you want to access the global Internet. The following command is modified: • **neighbor remove-private-as**

CHAPTER 15

Configuring BGP Neighbor Session Options

This module describes configuration tasks to configure various options involving Border Gateway Protocol (BGP) neighbor peer sessions. BGP is an interdomain routing protocol designed to provide loop-free routing between organizations. This module contains tasks that use BGP neighbor session commands to configure:

- Options to help an autonomous system migration
- TTL Security Check, a lightweight security mechanism to protect External BGP (eBGP) peering sessions from CPU-utilization-based attacks

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 Configuring BGP Neighbor Session Options

Before configuring advanced BGP features you should be familiar with the "Cisco BGP Overview" module and the "Configuring a Basic BGP Network" module.

Restrictions for Configuring BGP Neighbor Session Options

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 address family configurations.

Information About Configuring BGP Neighbor Session Options

BGP Neighbor Sessions

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. A BGP-speaking router does not discover another BGP-speaking device automatically. A network administrator usually manually configures the relationships between BGP-speaking routers.

A BGP neighbor device is a BGP-speaking router that has an active TCP connection to another BGP-speaking device. This relationship between BGP devices is often referred to as a peer instead of neighbor because a neighbor may imply the idea that the BGP devices are directly connected with no other router in between. Configuring BGP neighbor or peer sessions uses BGP neighbor session commands so this module uses the term “neighbor” over “peer.”

BFD Support of BGP IPv6 Neighbors

In Cisco IOS Release 15.1(2)S and later releases, Bidirectional Forwarding Detection (BFD) can be used to track fast forwarding path failure of BGP neighbors that have an IPv6 address. BFD is a detection protocol that is designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. BFD provides faster reconvergence time for BGP after a forwarding path failure.

TTL Security Check for BGP Neighbor Sessions

BGP Support for the TTL Security Check

When implemented for BGP, the TTL Security Check feature introduces a lightweight security mechanism to protect eBGP neighbor sessions from CPU utilization-based attacks. These types of attacks are typically brute force Denial of Service (DoS) attacks that attempt to disable the network by flooding the network with IP packets that contain forged source and destination IP addresses.

The TTL Security Check feature protects the eBGP neighbor session by comparing the value in the TTL field of received IP packets against a hop count that is configured locally for each eBGP neighbor session. If the value in the TTL field of the incoming IP packet is greater than or equal to the locally configured value, the IP packet is accepted and processed normally. If the TTL value in the IP packet is less than the locally configured value, the packet is silently discarded and no Internet Control Message Protocol (ICMP) message is generated. This is designed behavior; a response to a forged packet is unnecessary.

Although it is possible to forge the TTL field in an IP packet header, accurately forging the TTL count to match the TTL count from a trusted peer is impossible unless the network to which the trusted peer belongs has been compromised.

The TTL Security Check feature supports both directly connected neighbor sessions and multihop eBGP neighbor sessions. The BGP neighbor session is not affected by incoming packets that contain invalid TTL values. The BGP neighbor session will remain open, and the router will silently discard the invalid packet. The BGP session, however, can still expire if keepalive packets are not received before the session timer expires.

TTL Security Check for BGP Neighbor Sessions

The BGP Support for TTL Security Check feature is configured with the **neighbor ttl-security** command in router configuration mode or address family configuration mode. When this feature is enabled, BGP will establish or maintain a session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the peering session. Enabling this feature secures the eBGP session in the incoming direction only and has no effect on outgoing IP packets or the remote router. The *hop-count* argument is used to configure the maximum number of hops that separate the two peers. The TTL value is determined by the router from the configured hop count. The value for this argument is a number from 1 to 254.

TTL Security Check Support for Multihop BGP Neighbor Sessions

The BGP Support for TTL Security Check feature supports both directly connected neighbor sessions and multihop neighbor sessions. When this feature is configured for a multihop neighbor session, the **neighbor ebgp-multihop** router configuration command cannot be configured and is not needed to establish the neighbor session. These commands are mutually exclusive, and only one command is required to establish a multihop neighbor session. If you attempt to configure both commands for the same peering session, an error message will be displayed in the console.

To configure this feature for an existing multihop session, you must first disable the existing neighbor session with the **no neighbor ebgp-multihop** command. The multihop neighbor session will be restored when you enable this feature with the **neighbor ttl-security** command.

This feature should be configured on each participating router. To maximize the effectiveness of this feature, the *hop-count* argument should be strictly configured to match the number of hops between the local and external network. However, you should also consider path variation when configuring this feature for a multihop neighbor session.

Benefits of the BGP Support for TTL Security Check

The BGP Support for TTL Security Check feature provides an effective and easy-to-deploy solution to protect eBGP neighbor sessions from CPU utilization-based attacks. When this feature is enabled, a host cannot attack a BGP session if the host is not a member of the local or remote BGP network or if the host is not directly connected to a network segment between the local and remote BGP networks. This solution greatly reduces the effectiveness of DoS attacks against a BGP autonomous system.

BGP Support for TCP Path MTU Discovery per Session

Path MTU Discovery

The IP protocol family was designed to use a wide variety of transmission links. The maximum IP packet length is 65000 bytes. Most transmission links enforce a smaller maximum packet length limit, called the maximum transmission unit (MTU), which varies with the type of the transmission link. The design of IP accommodates link packet length limits by allowing intermediate routers to fragment IP packets as necessary for their outgoing links. The final destination of an IP packet is responsible for reassembling its fragments as necessary.

All TCP sessions are bounded by a limit on the number of bytes that can be transported in a single packet, and this limit is known as the maximum segment size (MSS). TCP breaks up packets into chunks in a transmit queue before passing packets down to the IP layer. A smaller MSS may not be fragmented at an IP device along the path to the destination device, but smaller packets increase the amount of bandwidth needed to transport the packets. The maximum TCP packet length is determined by both the MTU of the outbound interface on the source device and the MSS announced by the destination device during the TCP setup process.

Path MTU discovery (PMTUD) was developed as a solution to the problem of finding the optimal TCP packet length. PMTUD is an optimization (detailed in RFC 1191) wherein a TCP connection attempts to send the longest packets that will not be fragmented along the path from source to destination. It does this by using a flag, don't fragment (DF), in the IP packet. This flag is supposed to alter the behavior of an intermediate router that cannot send the packet across a link because it is too long. Normally the flag is off, and the router should fragment the packet and send the fragments. If a router tries to forward an IP datagram, with the DF bit set, to a link that has a lower MTU than the size of the packet, the router will drop the packet and return an ICMP Destination Unreachable message to the source of this IP datagram, with the code indicating "fragmentation needed and DF set." When the source device receives the ICMP message, it will lower the send MSS, and when TCP retransmits the segment, it will use the smaller segment size.

BGP Neighbor Session TCP PMTUD

TCP path MTU discovery is enabled by default for all BGP neighbor sessions, but there are situations when you may want to disable TCP path MTU discovery for one or all BGP neighbor sessions. Although PMTUD works well for larger transmission links (for example, Packet over Sonet links), a badly configured TCP implementation or a firewall may slow or stop the TCP connections from forwarding any packets. In this type of situation, you may need to disable TCP path MTU discovery.

In Cisco software, configuration options were introduced to permit TCP path MTU discovery to be disabled, or subsequently reenabled, either for a single BGP neighbor session or for all BGP sessions. To disable the TCP path MTU discovery globally for all BGP neighbors, use the **no bgp transport path-mtu-discovery** command in router configuration mode. To disable the TCP path MTU discovery for a single neighbor, use the **no neighbor transport path-mtu-discovery** command in router configuration mode or address family configuration mode. For more details, see the “Disabling TCP Path MTU Discovery Globally for All BGP Sessions” section or the “Disabling TCP Path MTU Discovery for a Single BGP Neighbor” section.

How to Configure BGP Neighbor Session Options

Configuring BFD for BGP IPv6 Neighbors

In Cisco IOS Release 15.1(2)S and later releases, Bidirectional Forwarding Detection (BFD) can be used for BGP neighbors that have an IPv6 address.

Once it has been verified that BFD neighbors are up, the **show bgp ipv6 unicast neighbors** command will indicate that BFD is being used to detect fast fallover on the specified neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **ipv6 cef**
5. **interface** *type number*
6. **ipv6 address** *ipv6-address* **/** *prefix-length*
7. **bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *multiplier-value*
8. **no shutdown**
9. **exit**
10. **router bgp** *autonomous-system-number*
11. **no bgp default ipv4-unicast**
12. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
13. **neighbor** *ipv6-address* **remote-as** *autonomous-system-number*
14. **neighbor** *ipv6-address* **fall-over bfd**
15. **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	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**ipv6 cef** **Example:** `Device(config)# ipv6 cef`	Enables Cisco Express Forwarding for IPv6.
Step 5	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/1`	Configures an interface type and number.

	Command or Action	Purpose
Step 6	**ipv6 address** *ipv6-address* / *prefix-length* **Example:** `Device(config-if)# ipv6 address 2001:DB8:1:1::1/64`	Configures an IPv6 address and enables IPv6 processing on an interface.
Step 7	**bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *multiplier-value* **Example:** `Device(config-if)# bfd interval 500 min_rx 500 multiplier 3`	Sets the baseline BFD session parameters on an interface.
Step 8	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts an interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and enters global configuration mode.
Step 10	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 11	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	Disables the default IPv4 unicast address family for establishing peering sessions. • We recommend configuring this command in the global scope.
Step 12	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6`	Enters address family configuration mode and enables IPv6 addressing.
Step 13	**neighbor** *ipv6-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 2001:DB8:2:1::4 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv6 BGP neighbor table of the local router.
Step 14	**neighbor** *ipv6-address* **fall-over bfd** **Example:**	Enables BGP to monitor the peering session of an IPv6 neighbor using BFD.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 2001:DB8:2:1::4 fall-over bfd`	
Step 15	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Configuring the TTL Security Check for BGP Neighbor Sessions

Perform this task to allow BGP to establish or maintain a session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the BGP neighbor session.

Before you begin

- To maximize the effectiveness of the BGP Support for TTL Security Check feature, we recommend that you configure it on each participating router. Enabling this feature secures the eBGP session in the incoming direction only and has no effect on outgoing IP packets or the remote router.

Note

- The **neighbor ebgp-multihop** command is not needed when the BGP Support for TTL Security Check feature is configured for a multihop neighbor session and should be disabled before configuring this feature.
- The effectiveness of the BGP Support for TTL Security Check feature is reduced in large-diameter multihop peerings. In the event of a CPU utilization-based attack against a BGP router that is configured for large-diameter peering, you may still need to shut down the affected neighbor sessions to handle the attack.
- This feature is not effective against attacks from a peer that has been compromised inside of the local and remote network. This restriction also includes peers that are on the network segment between the local and remote network.

SUMMARY STEPS

1. **enable**
2. **trace** [*protocol*] *destination*
3. **configure terminal**
4. **router bgp** *autonomous-system-number*
5. **neighbor** *ip-address* **ttl-security hops** *hop-count*
6. **end**
7. **show running-config**
8. **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	**trace** [*protocol*] *destination* **Example:** `Device# trace ip 10.1.1.1`	Discovers the routes of the specified protocol that packets will actually take when traveling to their destination. • Enter the **trace** command to determine the number of hops to the specified peer.
Step 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode, and creates a BGP routing process.
Step 5	**neighbor** *ip-address* **ttl-security hops** *hop-count* **Example:** `Device(config-router)# neighbor 10.1.1.1 ttl-security hops 2`	Configures the maximum number of hops that separate two peers. • The *hop-count* argument is set to the number of hops that separate the local and remote peer. If the expected TTL value in the IP packet header is 254, then the number 1 should be configured for the *hop-count* argument. The range of values is a number from 1 to 254. • When the BGP Support for TTL Security Check feature is enabled, BGP will accept incoming IP packets with a TTL value that is equal to or greater than the expected TTL value. Packets that are not accepted are discarded. • The example configuration sets the expected incoming TTL value to at least 253, which is 255 minus the TTL value of 2, and this is the minimum TTL value expected from the BGP peer. The local router will accept the peering session from the 10.1.1.1 neighbor only if it is one or two hops away.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

	Command or Action	Purpose
Step 7	**show running-config** **Example:** `Device# show running-config | begin bgp`	(Optional) Displays the contents of the currently running configuration file. • The output of this command displays the configuration of the **neighbor ttl-security** command for each peer under the BGP configuration section of output. That section includes the neighbor address and the configured hop count. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 8	**show ip bgp neighbors** [*ip-address*] **Example:** `Device# show ip bgp neighbors 10.4.9.5`	(Optional) Displays information about the TCP and BGP connections to neighbors. • This command displays "External BGP neighbor may be up to *number* hops away" when the BGP Support for TTL Security Check feature is enabled. The *number* value represents the hop count. It is a number from 1 to 254. **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 configuration of the BGP Support for TTL Security Check feature can be verified with the **show running-config** and **show ip bgp neighbors** commands. This feature is configured locally on each peer, so there is no remote configuration to verify.

The following is sample output from the **show running-config** command. The output shows that neighbor 10.1.1.1 is configured to establish or maintain the neighbor session only if the expected TTL count in the incoming IP packet is 253 or 254.

```
Router# show running-config
| begin bgp

router bgp 65000
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.1.1.1 remote-as 55000
 neighbor 10.1.1.1 ttl-security hops 2
 no auto-summary
.
.
.
```

The following is sample output from the **show ip bgp neighbors** command. The output shows that the local router will accept packets from the 10.1.1.1 neighbor if it is no more than 2 hops away. The configuration of this feature is displayed in the address family section of the output. The relevant line is shown in bold in the output.

```
Router# show ip bgp neighbors 10.1.1.1
BGP neighbor is 10.1.1.1,  remote AS 55000, external link
  BGP version 4, remote router ID 10.2.2.22
  BGP state = Established, up for 00:59:21
  Last read 00:00:21, 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:                  2          2
    Notifications:          0          0
    Updates:                0          0
    Keepalives:           226        227
    Route Refresh:          0          0
    Total:                228        229
  Default minimum time between advertisement runs is 5 seconds
 For address family: IPv4 Unicast
  BGP table version 1, neighbor version 1/0
  Output queue sizes : 0 self, 0 replicated
  Index 1, Offset 0, Mask 0x2
  Member of update-group 1
                                 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
  Number of NLRIs in the update sent: max 0, min 0
  Connections established 2; dropped 1
  Last reset 00:59:50, due to User reset
  External BGP neighbor may be up to 2 hops away.
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Local host: 10.2.2.22, Local port: 179
Foreign host: 10.1.1.1, Foreign port: 11001
Enqueued packets for retransmit: 0, input: 0  mis-ordered: 0 (0 bytes)
Event Timers (current time is 0xCC28EC):
Timer          Starts    Wakeups            Next
Retrans            63          0             0x0
TimeWait            0          0             0x0
AckHold            62         50             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0
iss:  712702676  snduna:  712703881  sndnxt:  712703881     sndwnd:  15180
irs: 2255946817  rcvnxt: 2255948041  rcvwnd:      15161  delrcvwnd:   1223
SRTT: 300 ms, RTTO: 607 ms, RTV: 3 ms, KRTT: 0 ms
minRTT: 0 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: passive open, nagle, gen tcbs

Datagrams (max data segment is 1460 bytes):
```

```
Rcvd: 76 (out of order: 0), with data: 63, total data bytes: 1223
Sent: 113 (retransmit: 0, fastretransmit: 0), with data: 62, total data bytes: 4
```

Configuring BGP Support for TCP Path MTU Discovery per Session

This section contains the following tasks:

Disabling TCP Path MTU Discovery Globally for All BGP Sessions

Perform this task to disable TCP path MTU discovery for all BGP sessions. TCP path MTU discovery is enabled by default when you configure BGP sessions, but we recommend that you enter the **show ip bgp neighbors** command to ensure that TCP path MTU discovery is enabled.

Before you begin

This task assumes that you have previously configured BGP neighbors with active TCP connections.

SUMMARY STEPS

1. **enable**
2. **show ip bgp neighbors** [*ip-address*]
3. **configure terminal**
4. **router bgp** *autonomous-system-number*
5. **no bgp transport path-mtu-discovery**
6. **end**
7. **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	**show ip bgp neighbors** [*ip-address*] **Example:** `Device# show ip bgp neighbors`	(Optional) Displays information about the TCP and BGP connections to neighbors. • Use this command to determine whether BGP neighbors have TCP path MTU discovery enabled. **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 3	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 50000`	Enters router configuration mode to create or configure a BGP routing process.
Step 5	**no bgp transport path-mtu-discovery** **Example:** `Device(config-router)# no bgp transport path-mtu-discovery`	Disables TCP path MTU discovery for all BGP sessions.
Step 6	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.
Step 7	**show ip bgp neighbors** [*ip-address*] **Example:** `Device# show ip bgp neighbors`	(Optional) Displays information about the TCP and BGP connections to neighbors. • In this example, the output from this command will not display that any neighbors have TCP path MTU enabled. **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 neighbors** command shows that TCP path MTU discovery is enabled for BGP neighbors. Two entries in the output—Transport(tcp) path-mtu-discovery is enabled and path mtu capable—show that TCP path MTU discovery is enabled.

```
Router# show ip bgp neighbors

BGP neighbor is 172.16.1.2,  remote AS 45000, internal link
  BGP version 4, remote router ID 172.16.1.99
.
.
.
 For address family: IPv4 Unicast
  BGP table version 5, neighbor version 5/0
.
.
.
  Address tracking is enabled, the RIB does have a route to 172.16.1.2
  Address tracking requires at least a /24 route to the peer
  Connections established 3; dropped 2
  Last reset 00:00:35, due to Router ID changed
  Transport(tcp) path-mtu-discovery is enabled
.
```

```
.
.
SRTT: 146 ms, RTTO: 1283 ms, RTV: 1137 ms, KRTT: 0 ms
minRTT: 8 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: higher precedence, retransmission timeout, nagle, path mtu capable
```

The following is sample output from the **show ip bgp neighbors** command after the **no bgp transport path-mtu-discovery** command has been entered. Note that the path mtu entries are missing.

```
Router# show ip bgp neighbors

BGP neighbor is 172.16.1.2,  remote AS 45000, internal link
  BGP version 4, remote router ID 172.16.1.99
.
.
.
 For address family: IPv4 Unicast
  BGP table version 5, neighbor version 5/0
.
.
.
  Address tracking is enabled, the RIB does have a route to 172.16.1.2
  Address tracking requires at least a /24 route to the peer
  Connections established 3; dropped 2
  Last reset 00:00:35, due to Router ID changed
.
.
.
SRTT: 146 ms, RTTO: 1283 ms, RTV: 1137 ms, KRTT: 0 ms
minRTT: 8 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: higher precedence, retransmission timeout, nagle
```

Disabling TCP Path MTU Discovery for a Single BGP Neighbor

Perform this task to establish a peering session with an internal BGP (iBGP) neighbor and then disable TCP path MTU discovery for the BGP neighbor session. The **neighbor transport** command can be used in router configuration mode or address family configuration mode.

Before you begin

This task assumes that you know that TCP path MTU discovery is enabled by default for all your BGP neighbors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
5. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** {*ip-address*| *peer-group-name*} **activate**
7. **no neighbor** {*ip-address*| *peer-group-name*} **transport**{**connection-mode** | **path-mtu-discovery**}
8. **end**
9. **show ip bgp neighbors**

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	**address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]} **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations. • The example creates an IPv4 unicast address family session.
Step 5	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 6	**neighbor** {*ip-address*| *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 172.16.1.1 activate`	Activates the neighbor under the IPv4 address family.
Step 7	**no neighbor** {*ip-address*| *peer-group-name*} **transport**{**connection-mode** | **path-mtu-discovery**} **Example:** `Device(config-router-af)# no neighbor 172.16.1.1 transport path-mtu-discovery`	Disables TCP path MTU discovery for a single BGP neighbor. • In this example, TCP path MTU discovery is disabled for the neighbor at 172.16.1.1.
Step 8	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 9	**show ip bgp neighbors** **Example:** `Device# show ip bgp neighbors`	(Optional) Displays information about the TCP and BGP connections to neighbors. • In this example, the output from this command will not display that the neighbor has TCP path MTU discovery enabled. **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 shows that TCP path MTU discovery has been disabled for BGP neighbor 172.16.1.1 but that it is still enabled for BGP neighbor 192.168.2.2. Two entries in the output—Transport(tcp) path-mtu-discovery is enabled and path mtu capable—show that TCP path MTU discovery is enabled.

```
Router# show ip bgp neighbors
BGP neighbor is 172.16.1.1,  remote AS 45000, internal link
  BGP version 4, remote router ID 172.17.1.99
.
.
.
  Address tracking is enabled, the RIB does have a route to 172.16.1.1
  Address tracking requires at least a /24 route to the peer
  Connections established 1; dropped 0
  Last reset never
.
.
.
SRTT: 165 ms, RTTO: 1172 ms, RTV: 1007 ms, KRTT: 0 ms
minRTT: 20 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: higher precedence, retransmission timeout, nagle
.
.
.
BGP neighbor is 192.168.2.2,  remote AS 50000, external link
  BGP version 4, remote router ID 10.2.2.99
.
.
.
 For address family: IPv4 Unicast
  BGP table version 4, neighbor version 4/0
.
.
.
  Address tracking is enabled, the RIB does have a route to 192.168.2.2
  Address tracking requires at least a /24 route to the peer
  Connections established 2; dropped 1
  Last reset 00:05:11, due to User reset
  Transport(tcp) path-mtu-discovery is enabled
.
.
.
SRTT: 210 ms, RTTO: 904 ms, RTV: 694 ms, KRTT: 0 ms
```

```
minRTT: 20 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: higher precedence, retransmission timeout, nagle, path mtu capable
```

Enabling TCP Path MTU Discovery Globally for All BGP Sessions

Perform this task to enable TCP path MTU discovery for all BGP sessions. TCP path MTU discovery is enabled by default when you configure BGP sessions, but if the BGP Support for TCP Path MTU Discovery per Session feature has been disabled, you can use this task to reenable it. To verify that TCP path MTU discovery is enabled, use the **show ip bgp neighbors** command.

Before you begin

This task assumes that you have previously configured BGP neighbors with active TCP connections.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp transport path-mtu-discovery**
5. **end**
6. **show ip bgp neighbors**

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 to create or configure a BGP routing process.
Step 4	**bgp transport path-mtu-discovery** **Example:** `Device(config-router)# bgp transport path-mtu-discovery`	Enables TCP path MTU discovery for all BGP sessions.
Step 5	**end** **Example:**	Exits router configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config-router)# end`	
Step 6	**show ip bgp neighbors** **Example:** `Device# show ip bgp neighbors`	(Optional) Displays information about the TCP and BGP connections to neighbors. • In this example, the output from this command will show that all neighbors have TCP path MTU discovery enabled. **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 neighbors** command shows that TCP path MTU discovery is enabled for BGP neighbors. Two entries in the output—Transport(tcp) path-mtu-discovery is enabled and path mtu capable—show that TCP path MTU discovery is enabled.

```
Router# show ip bgp neighbors
BGP neighbor is 172.16.1.2,  remote AS 45000, internal link
  BGP version 4, remote router ID 172.16.1.99
.
.
.
 For address family: IPv4 Unicast
  BGP table version 5, neighbor version 5/0
.
.
.
  Address tracking is enabled, the RIB does have a route to 172.16.1.2
  Address tracking requires at least a /24 route to the peer
  Connections established 3; dropped 2
  Last reset 00:00:35, due to Router ID changed
  Transport(tcp) path-mtu-discovery is enabled
.
.
.
SRTT: 146 ms, RTTO: 1283 ms, RTV: 1137 ms, KRTT: 0 ms
minRTT: 8 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: higher precedence, retransmission timeout, nagle, path mtu capable
```

Enabling TCP Path MTU Discovery for a Single BGP Neighbor

Perform this task to establish a peering session with an eBGP neighbor and then enable TCP path MTU discovery for the BGP neighbor session. The **neighbor transport** command can be used in router configuration mode or address family configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **router bgp** *autonomous-system-number*
4. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
5. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** {*ip-address*| *peer-group-name*} **activate**
7. **neighbor** {*ip-address*| *peer-group-name*} **transport**{**connection-mode** | **path-mtu-discovery**}
8. **end**
9. **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 45000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family** {**ipv4** [**mdt**	**multicast**	**unicast** [**vrf** *vrf-name*]	**vrf** *vrf-name*]	**vpnv4** [**unicast**]} **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations. • The example creates an IPv4 unicast address family session.
Step 5	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.2.2 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 6	**neighbor** {*ip-address*| *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.2 activate`	Activates the neighbor under the IPv4 address family.
Step 7	**neighbor** {*ip-address*	*peer-group-name*} **transport**{**connection-mode**	**path-mtu-discovery**} **Example:**	Enables TCP path MTU discovery for a single BGP neighbor.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 192.168.2.2 transport path-mtu-discovery`	
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*] **Example:** `Device# 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 neighbors** command shows that TCP path MTU discovery is enabled for the BGP neighbor at 192.168.2.2. Two entries in the output—Transport(tcp) path-mtu-discovery is enabled and path-mtu capable—show that TCP path MTU discovery is enabled.

```
Router# show ip bgp neighbors 192.168.2.2
BGP neighbor is 192.168.2.2,  remote AS 50000, external link
  BGP version 4, remote router ID 10.2.2.99
.
.
.
 For address family: IPv4 Unicast
  BGP table version 4, neighbor version 4/0
.
.
.
  Address tracking is enabled, the RIB does have a route to 192.168.2.2
  Address tracking requires at least a /24 route to the peer
  Connections established 2; dropped 1
  Last reset 00:05:11, due to User reset
  Transport(tcp) path-mtu-discovery is enabled
.
.
.
SRTT: 210 ms, RTTO: 904 ms, RTV: 694 ms, KRTT: 0 ms
minRTT: 20 ms, maxRTT: 300 ms, ACK hold: 200 ms
Flags: higher precedence, retransmission timeout, nagle, path mtu capable
```

Configuration Examples for BGP Neighbor Session Options

Example: Configuring BFD for a BGP IPv6 Neighbor

The following example configures FastEthernet interface 0/1 with the IPv6 address 2001:DB8:4:1::1. Bidirectional Forwarding Detection (BFD) is configured for the BGP neighbor at 2001:DB8:5:1::2. BFD will track forwarding path failure of the BGP neighbor and provide faster reconvergence time for BGP after a forwarding path failure.

```
ipv6 unicast-routing
ipv6 cef
interface fastethernet 0/1
 ipv6 address 2001:DB8:4:1::1/64
 bfd interval 500 min_rx 500 multiplier 3
 no shutdown
 exit
router bgp 65000
 no bgp default ipv4-unicast
 address-family ipv6 unicast
 neighbor 2001:DB8:5:1::2 remote-as 65001
 neighbor 2001:DB8:5:1::2 fall-over bfd
 end
```

Example: Configuring the TTL-Security Check

The example configurations in this section show how to configure the BGP Support for TTL Security Check feature.

The following example uses the **trace** command to determine the hop count to an eBGP peer. The hop count number is displayed in the output for each networking device that IP packets traverse to reach the specified neighbor. In the following example, the hop count for the 10.1.1.1 neighbor is 1.

```
Router# trace ip 10.1.1.1
Type escape sequence to abort.
Tracing the route to 10.1.1.1
  1 10.1.1.1 0 msec *  0 msec
```

The following example sets the hop count to 2 for the 10.1.1.1 neighbor. Because the hop-count argument is set to 2, BGP will accept only IP packets with a TTL count in the header that is equal to or greater than 253.

```
Router(config-router)# neighbor 10.1.1.1 ttl-security hops 2
```

Examples: Configuring BGP Support for TCP Path MTU Discovery per Session

This section contains the following configuration examples:

Example: Disabling TCP Path MTU Discovery Globally for All BGP Sessions

The following example shows how to disable TCP path MTU discovery for all BGP neighbor sessions. Use the **show ip bgp neighbors** command to verify that TCP path MTU discovery has been disabled.

```
enable
configure terminal
 router bgp 45000
  no bgp transport path-mtu-discovery
  end
show ip bgp neighbors
```

Example: Disabling TCP Path MTU Discovery for a Single BGP Neighbor

The following example shows how to disable TCP path MTU discovery for an eBGP neighbor at 192.168.2.2:

```
enable
configure terminal
 router bgp 45000
  neighbor 192.168.2.2 remote-as 50000
  neighbor 192.168.2.2 activate
  no neighbor 192.168.2.2 transport path-mtu-discovery
  end
show ip bgp neighbors 192.168.2.2
```

Example: Enabling TCP Path MTU Discovery Globally for All BGP Sessions

The following example shows how to enable TCP path MTU discovery for all BGP neighbor sessions. Use the **show ip bgp neighbors** command to verify that TCP path MTU discovery has been enabled.

```
enable
configure terminal
 router bgp 45000
  bgp transport path-mtu-discovery
  end
show ip bgp neighbors
```

Example: Enabling TCP Path MTU Discovery for a Single BGP Neighbor

The following example shows how to enable TCP path MTU discovery for an eBGP neighbor at 192.168.2.2. Use the **show ip bgp neighbors** command to verify that TCP path MTU discovery has been enabled.

```
enable
configure terminal
 router bgp 45000
  neighbor 192.168.2.2 remote-as 50000
  neighbor 192.168.2.2 activate
  neighbor 192.168.2.2 transport path-mtu-discovery
  end
show ip bgp neighbors 192.168.2.2
```

Additional References

Related Documents

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

Related Topic	Document Title
BGP commands: complete command syntax, command mode, defaults, command history, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
Overview of Cisco BGP conceptual information with links to all the individual BGP modules	"Cisco BGP Overview" module
Conceptual and configuration details for basic BGP tasks	"Configuring a Basic BGP Network" module
Conceptual and configuration details for advanced BGP tasks	"Configuring Advanced BGP Features" module
Bidirectional Forwarding Detection configuration tasks	*IP Routing: BFD Configuration Guide*

Standards

Standard	Title
MDT SAFI	MDT SAFI

MIBs

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

RFCs

RFC	Title
RFC 1191	*Path MTU Discovery*
RFC 1771	*A Border Gateway Protocol 4 (BGP-4)*
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 of an Autonomous System (AS)*
RFC 2858	*Multiprotocol Extensions for BGP-4*
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 Configuring BGP Neighbor Session Options

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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 28: Feature Information for Configuring BGP Neighbor Session Options Features

Feature Name	Releases	Feature Information
BGP Support for TCP Path MTU Discovery per Session	12.2(33)SRA 12.2(31)SB 12.2(33)SXH 12.4(20)T 15.0(1)S	BGP support for TCP path maximum transmission unit (MTU) discovery introduced the ability for BGP to automatically discover the best TCP path MTU for each BGP session. The TCP path MTU is enabled by default for all BGP neighbor sessions, but you can disable, and subsequently enable, the TCP path MTU globally for all BGP sessions or for an individual BGP neighbor session. The following commands were introduced or modified by this feature: **bgp transport**, **neighbor transport**, **show ip bgp neighbors**.
BGP Support for TTL Security Check	12.0(27)S 12.3(7)T 12.2(25)S 12.2(18)SXE 15.0(1)S	The BGP Support for TTL Security Check feature introduced a lightweight security mechanism to protect external Border Gateway Protocol (eBGP) peering sessions from CPU utilization-based attacks using forged IP packets. Enabling this feature prevents attempts to hijack the eBGP peering session by a host on a network segment that is not part of either BGP network or by a host on a network segment that is not between the eBGP peers. The following commands were introduced or modified by this feature: **neighbor ttl-security**, **show ip bgp neighbors**.

Feature Name	Releases	Feature Information
BGP IPv6 Client for Single-Hop BFD	15.1(2)S 15.2(3)T 15.2(4)S	Bidirectional Forwarding Detection (BFD) can be used to track fast forwarding path failure of BGP neighbors that use an IPv6 address. The following command was modified by this feature: **neighbor fall-over**. In Cisco IOS Release 15.2(4)S, support was added for the Cisco 7200 series router.

CHAPTER 16

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
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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 29: Feature Information for BGP Neighbor Policy

Feature Name	Releases	Feature Information
BGP Neighbor Policy	12.4(11)T	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. The following commands were modified: **show ip bgp neighbors**, and **show ip bgp template peer-policy**.

CHAPTER 17

BGP Dynamic Neighbors

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

Finding Feature Information

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

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

Information About BGP Dynamic Neighbors

BGP Dynamic Neighbors

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

In Cisco IOS XE Denali 16.3 release, support for BGP dynamic neighbors was extended to IPv6 BGP peering with VRF support.

After a subnet range is configured for a BGP peer group and a TCP session is initiated by another router for an IP address in the subnet range, a new BGP neighbor is dynamically created as a member of that group.

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

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

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

How to Configure BGP Dynamic Neighbors

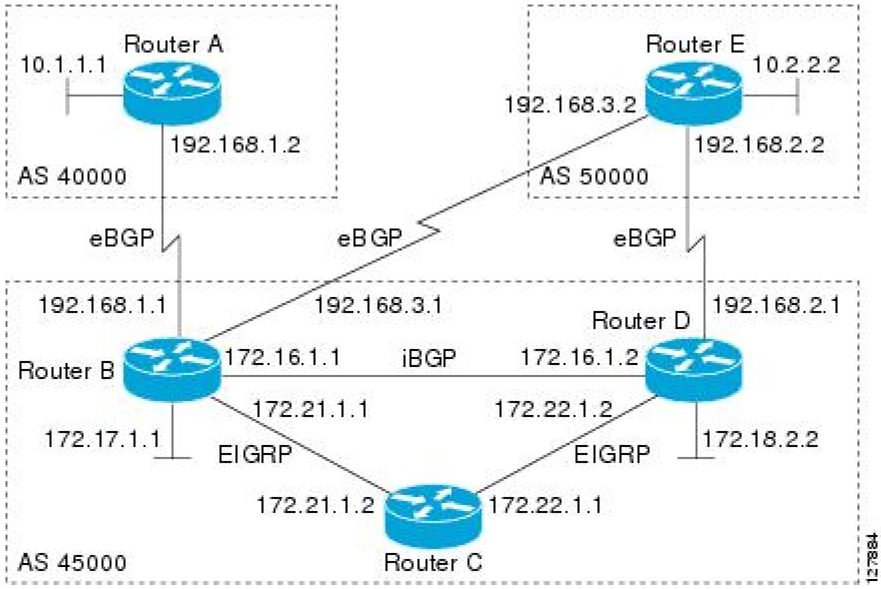
Implementing BGP Dynamic Neighbors Using Subnet Ranges

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

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

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

Figure 34: BGP Dynamic Neighbor Topology



Before you begin

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

Note This task supports only IPv4 BGP peering.

SUMMARY STEPS

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

17. **neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]
18. Return to the first router.
19. **show ip bgp summary**
20. **show ip bgp peer-group** [*peer-group-name*] [**summary**]
21. **show ip bgp neighbors** [*ip-address*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `DeviceB> enable`	Enables privileged EXEC mode. • Enter your password if prompted. • The configuration is entered on router B.
Step 2	**configure terminal** **Example:** `DeviceB# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `DeviceB(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**bgp log-neighbor-changes** **Example:** `DeviceB(config-router)# bgp log-neighbor-changes`	(Optional) Enables logging of BGP neighbor status changes (up or down) and neighbor resets. • Use this command for troubleshooting network connectivity problems and measuring network stability. Unexpected neighbor resets might indicate high error rates or high packet loss in the network and should be investigated.
Step 5	**neighbor** *peer-group-name* **peer-group** **Example:** `DeviceB(config-router)# neighbor group192 peer-group`	Creates a BGP peer group. • In this example, a peer group named group192 is created. This group will be used as a listen range group.
Step 6	**bgp listen** [**limit** *max-number*] **Example:** `DeviceB(config-router)# bgp listen limit 200`	Sets a global limit of BGP dynamic subnet range neighbors. • Use the optional **limit** keyword and *max-number* argument to define the maximum number of BGP dynamic subnet range neighbors that can be created. **Note** Only the syntax applicable to this task is used in this example. For the complete syntax, see Step 7.

	Command or Action	Purpose
Step 7	**bgp listen** [**limit** *max-number* | **range** *network* / *length* **peer-group** *peer-group-name*] **Example:** `DeviceB(config-router)# bgp listen range 192.168.0.0/16 peer-group group192`	Associates a subnet range with a BGP peer group and activates the BGP dynamic neighbors feature. • Use the optional **limit** keyword and *max-number* argument to define the maximum number of BGP dynamic neighbors that can be created. • Use the optional **range** keyword and *network* / *length* argument to define a prefix range to be associated with the specified peer group. • In this example, the prefix range 192.168.0.0/16 is associated with the listen range group named group192.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `DeviceB(config-router)# neighbor group192 ebgp-multihop 255`	Accepts and attempts BGP connections to external peers residing on networks that are not directly connected.
Step 9	**neighbor** *peer-group-name* **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `DeviceB(config-router)# neighbor group192 remote-as 40000 alternate-as 50000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • Use the optional **alternate-as** keyword and *autonomous-system-number* argument to identify up to five alternate autonomous system numbers for listen range neighbors. • In this example, the peer group named group192 is configured with two possible autonomous system numbers. **Note** The **alternate-as** keyword is used only with the listen range peer groups, not with individual BGP neighbors.
Step 10	**address-family ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]] **Example:** `DeviceB(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations.
Step 11	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `DeviceB(config-router-af)# neighbor group192 activate`	Activates the neighbor or listen range peer group for the configured address family. • In this example, the neighbor 172.16.1.1 is activated for the IPv4 address family.

	Command or Action	Purpose
		Note Usually BGP peer groups cannot be activated using this command, but the listen range peer groups are a special case.
Step 12	**end** **Example:** `DeviceB(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 13	Move to another router that has an interface within the subnet range for the BGP peer group configured in this task.	—
Step 14	**enable** **Example:** `DeviceE> enable`	Enables privileged EXEC mode. • Enter your password if prompted. • The configuration is entered on Router E.
Step 15	**configure terminal** **Example:** `DeviceE# configure terminal`	Enters global configuration mode.
Step 16	**router bgp** *autonomous-system-number* **Example:** `DeviceE(config)# router bgp 50000`	Enters router configuration mode for the specified routing process.
Step 17	**neighbor** {*ip-address*| *peer-group-name*} **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `DeviceE(config-router)# neighbor 192.168.3.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • In this example, the interface (192.168.3.2 in the figure above) at Router E is with the subnet range set for the BGP listen range group, group192. When TCP opens a session to peer to Router B, Router B creates this peer dynamically.
Step 18	Return to the first router.	—
Step 19	**show ip bgp summary** **Example:** `DeviceB# show ip bgp summary`	(Optional) Displays the BGP path, prefix, and attribute information for all connections to BGP neighbors. • In this step, the configuration has returned to Router B.
Step 20	**show ip bgp peer-group** [*peer-group-name*] [**summary**] **Example:**	(Optional) Displays information about BGP peer groups.

	Command or Action	Purpose
	`DeviceB# show ip bgp peer-group group192`	
Step 21	**show ip bgp neighbors** [*ip-address*] **Example:** `DeviceB# show ip bgp neighbors 192.168.3.2`	(Optional) Displays information about BGP and TCP connections to neighbors. • In this example, information is displayed about the dynamically created neighbor at 192.168.3.2. The IP address of this BGP neighbor can be found in the output of either the **show ip bgp summary** or the **show ip bgp peer-group** command. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Examples

The following output examples were taken from Router B in the figure above after the appropriate configuration steps in this task were completed on both Router B and Router E.

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

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

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

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

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

```
Router# show ip bgp neighbors 192.168.3.2
BGP neighbor is *192.168.3.2,  remote AS 50000, external link
 Member of peer-group group192 for session parameters
 Belongs to the subnet range group: 192.168.0.0/16
  BGP version 4, remote router ID 192.168.3.2
  BGP state = Established, up for 00:06:35
  Last read 00:00:33, last write 00:00:25, hold time is 180, keepalive intervals
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Address family IPv4 Unicast: advertised and received
  Message statistics:
    InQ depth is 0
    OutQ depth is 0

                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                0          0
    Keepalives:             7          7
    Route Refresh:          0          0
    Total:                  8          8
  Default minimum time between advertisement runs is 30 seconds
 For address family: IPv4 Unicast
  BGP table version 1, neighbor version 1/0
  Output queue size : 0
  Index 1, Offset 0, Mask 0x2
  1 update-group member
  group192 peer-group member
.
.
.
```

Configuring BGP IPv6 Dynamic Neighbor Support with VRF Support

In Cisco IOS XE Denali 16.3 release, support for BGP dynamic neighbors was extended to IPv6 BGP peering.

Note You can also configure BGP IPv6 dynamic neighbors without VRF support.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **bgp listen** [**limit** *max-number* | **range** *network* / *length* **peer-group** *peer-group-name*]
5. **address-family** [**ipv4** | **ipv6**] [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]]
6. **bgp listen** [**limit** *max-number*]
7. **neighbor** *peer-group-name* **peer-group**
8. **neighbor** *peer-group-name* **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*]

9. **address-family** [**ipv4** | **ipv6**] [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]]
10. **neighbor** {*ip-address* | *peer-group-name*} **activate**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted. • The configuration is entered on router B.
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 listen** [**limit** *max-number* | **range** *network* / *length* **peer-group** *peer-group-name*] **Example:** `Device(config-router)# bgp listen range 2001::0/64 peer-group group192`	Associates a subnet range with a BGP peer group and activates the BGP dynamic neighbors feature. • Use the optional **limit** keyword and *max-number* argument to define the maximum number of BGP dynamic neighbors that can be created. • Use the optional **range** keyword and *network* / *length* argument to define a prefix range to be associated with the specified peer group. • In this example, the prefix range 2001::0/64 is associated with the listen range group named group192.
Step 5	**address-family** [**ipv4** | **ipv6**] [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]] **Example:** `Device(config-router-af)# address-family ipv6 unicast vrf vrf1`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations.
Step 6	**bgp listen** [**limit** *max-number*] **Example:** `Device(config-router)# bgp listen limit 500`	Specifies the maximum number of prefixes in VRF address family.
Step 7	**neighbor** *peer-group-name* **peer-group**	Creates a BGP peer group.

	Command or Action	Purpose
	Example: `Device(config-router)# neighbor group192 peer-group`	• In this example, a peer group named group192 is created. This group will be used as a listen range group.
Step 8	**neighbor** *peer-group-name* **remote-as** *autonomous-system-number* [**alternate-as** *autonomous-system-number...*] **Example:** `Device(config-router)# neighbor group192 remote-as 101 alternate-as 102`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv6 BGP neighbor table. • Use the optional **alternate-as** keyword and *autonomous-system-number* argument to identify up to five alternate autonomous system numbers for listen range neighbors. • In this example, the peer group named group192 is configured with two possible autonomous system numbers. **Note** The **alternate-as** keyword is used only with the listen range peer groups, not with individual BGP neighbors.
Step 9	**address-family** [**ipv4** | **ipv6**] [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*]] **Example:** `Device(config-router-af)# address-family ipv4 unicast vrf vrf1`	Enable IPv4 address family for this peer-group.
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor group192 activate`	Activates the neighbor or listen range peer group for the configured address family.
Step 11	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Verifying BGP IPv6 Dynamic Neighbor Configuration

Use the **show bgp ipv6 unicast summary** command to verify the BGP IPv6 unicast address family configuration in global routing table:

```
Device# show bgp ipv6 unicast summary
BGP router identifier 192.168.3.1, local AS number 45000
BGP table version is 1, main routing table version 1
Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd
*2001::1 4 50000 2 2 0 0 0 00:00:37 0
* Dynamically created based on a listen range command
```

```
Dynamically created neighbors: 1/(200 max), Subnet ranges: 1
BGP peergroup group192 listen range group members:
2001::0/64
```

Use the **show bgp { ipv4 | ipv6 } unicast peer-group< name>** command to verify the IPv6 dynamic neighbors configuration in global routing table:

```
Device# show bgp ipv6 unicast peer-group group192
BGP peer-group is group192, remote AS 40000
BGP peergroup group192 listen range group members:
2001::0/64
BGP version 4
Default minimum time between advertisement runs is 30 seconds
For address family: IPv6 Unicast
BGP neighbor is group192, peer-group external, members:
*2001::1
Index 0, Offset 0, Mask 0x0
Update messages formatted 0, replicated 0
Number of NLRIs in the update sent: max 0, min 0
```

You can use the following commands to verify the BGP IPv6 dynamic neighbors configuration in the VRF routing table:

- **show bgp vpnv6 unicast vrf <name> neighbors**
- **show bgp vpnv6 unicast vrf <name> summary**
- **show bgp vpnv6 unicast vrf <name> peer-group <name>**
- **debug bgp [ipv6 | vpnv6] unicast range**

Configuration Examples for BGP Dynamic Neighbors

Example: Implementing BGP Dynamic Neighbors Using Subnet Ranges

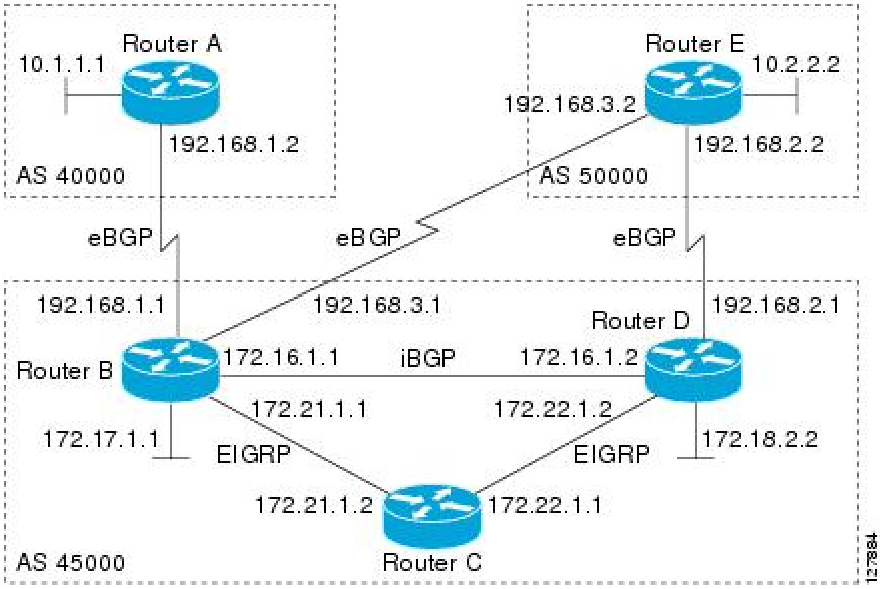
In the following example, two BGP peer groups are created on Router B in the figure below, a global limit is set on the number of dynamic BGP neighbors, and a subnet range is associated with a peer group. Configuring the subnet range enables the dynamic BGP neighbor process. The peer groups are added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

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

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

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

Figure 35: BGP Dynamic Neighbor Topology



Router B

```
enable
configure terminal
router bgp 45000
 bgp log-neighbor-changes
 bgp listen limit 200
 bgp listen range 172.21.0.0/16 peer-group group172
 bgp listen range 192.168.0.0/16 peer-group group192
 neighbor group172 peer-group
 neighbor group172 remote-as 45000
 neighbor group192 peer-group
 neighbor group192 remote-as 40000 alternate-as 50000
 neighbor 172.16.1.2 remote-as 45000
 address-family ipv4 unicast
 neighbor group172 activate
 neighbor group192 activate
 neighbor 172.16.1.2 activate
 end
```

Router A

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

Router E

```
enable
configure terminal
router bgp 50000
```

```
 neighbor 192.168.3.1 remote-as 45000
 exit
```

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

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

Example: Configuring BGP IPv6 Dynamic Neighbor Support with VRF Support

Configuring BGP IPv6 Dynamic Neighbor Support with VRF Support

```
enable
configure terminal
router bgp 55000
 bgp listen range 2001::0/64 peer-group group182
  address-family ipv6 unicast vrf vrf2
  bgp listen limit 600
  neighbor group182 peer-group
  neighbor group182 remote-as 103 alternate-as 104
  address-family ipv4 unicast vrf vrf2
  neighbor group182 activate
 end
```

Configuring BGP IPv6 Dynamic Neighbor Support without VRF Support

```
enable
configure terminal
router bgp 100
 bgp listen range 2001::0/64 peer-group group192
 bgp listen limit 500
 neighbor group192 peer-group
 neighbor group192 remote-as 101 alternate-as 102
 address family ipv6 unicast
  neighbor group192 activate
  address family ipv4 unicast
  neighbor group192 activate
 end
```

Additional References

Related Documents

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

Standards and RFCs

Standard/RFC	Title
RFC 2918	*Route Refresh Capability for BGP-4*

Technical Assistance

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

Feature Information for BGP Dynamic Neighbors

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

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

Table 30: Feature Information for BGP Dynamic Neighbors

Feature Name	Releases	Feature Information
BGP Dynamic Neighbors	15.1(2)T	BGP dynamic neighbor support allows BGP peering to a group of remote neighbors that are defined by a range of IP addresses. Each range can be configured as a subnet IP address. BGP dynamic neighbors are configured using a range of IP addresses and BGP peer groups. After a subnet range is configured for a BGP peer group and a TCP session is initiated for an IP address in the subnet range, a new BGP neighbor is dynamically created as a member of that group. The new BGP neighbor will inherit any configuration for the peer group. The following commands were introduced or modified by this feature: **bgp listen**, **debug ip bgp range**, **neighbor remote-as**, **show ip bgp neighbors**, **show ip bgp peer-group**, and **show ip bgp summary**.
BGP IPv6 Dynamic Neighbor Support and VRF Support	Cisco IOS XE Denali 16.3.1	In Cisco IOS XE Denali 16.3 release, support for BGP dynamic neighbors was extended to IPv6 BGP peering with support for VRF. The following commands were introduced or modified by this feature: **bgp listen**, **debug ip bgp range**, **neighbor remote-as**, **show bgp neighbors**, **show bgp summary**, **show bgp vpnv6 unicast vrf neighbors**, **show bgp vpnv6 unicast vrf peer-group** , **show bgp vpnv6 unicast vrf summary**.

CHAPTER 18

Configuring Internal BGP Features

This module describes how to configure internal Border Gateway Protocol (BGP) features. Internal BGP (iBGP) refers to running BGP on networking devices within one autonomous system. BGP is an interdomain routing protocol designed to provide loop-free routing between separate routing domains (autonomous systems) that contain independent routing policies. Many companies now have large internal networks, and there are many issues involved in scaling the existing internal routing protocols to match the increasing traffic demands while maintaining network efficiency.

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 Internal BGP Features

BGP Routing Domain Confederation

One way to reduce the internal BGP (iBGP) mesh is to divide an autonomous system into multiple subautonomous systems and group them into a single confederation. To the outside world, the confederation looks like a single autonomous system. Each autonomous system is fully meshed within itself and has a few connections to other autonomous systems in the same confederation. Even though the peers in different autonomous systems have external BGP (eBGP) sessions, they exchange routing information as if they were iBGP peers. Specifically, the next hop, Multi Exit Discriminator (MED) attribute, and local preference information are preserved. This feature allows the you to retain a single Interior Gateway Protocol (IGP) for all of the autonomous systems.

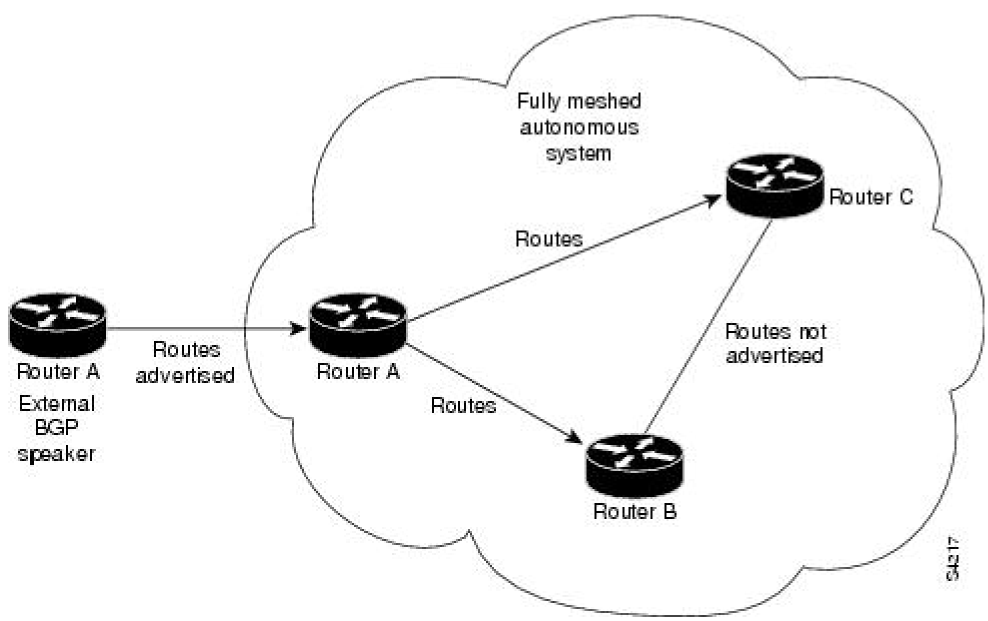
To configure a BGP confederation, you must specify a confederation identifier. To the outside world, the group of autonomous systems will look like a single autonomous system with the confederation identifier as the autonomous system number.

BGP Route Reflector

BGP requires that all iBGP speakers be fully meshed. However, this requirement does not scale well when there are many iBGP speakers. Instead of configuring a confederation, another way to reduce the iBGP mesh is to configure a route reflector.

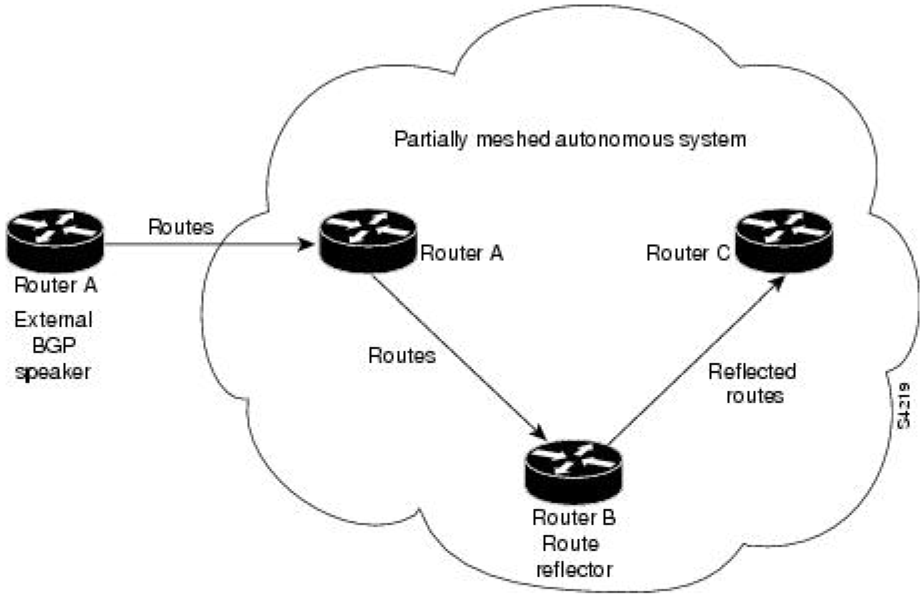
The figure below illustrates a simple iBGP configuration with three iBGP speakers (Routers A, B, and C). Without route reflectors, when Router A receives a route from an external neighbor, it must advertise it to both routers B and C. Routers B and C do not readvertise the iBGP learned route to other iBGP speakers because the routers do not pass on routes learned from internal neighbors to other internal neighbors, thus preventing a routing information loop.

Figure 36: Three Fully Meshed iBGP Speakers



With route reflectors, all iBGP speakers need not be fully meshed because there is a method to pass learned routes to neighbors. In this model, an iBGP peer is configured to be a route reflector responsible for passing iBGP learned routes to a set of iBGP neighbors. In the figure below, Router B is configured as a route reflector. When the route reflector receives routes advertised from Router A, it advertises them to Router C, and vice versa. This scheme eliminates the need for the iBGP session between Routers A and C.

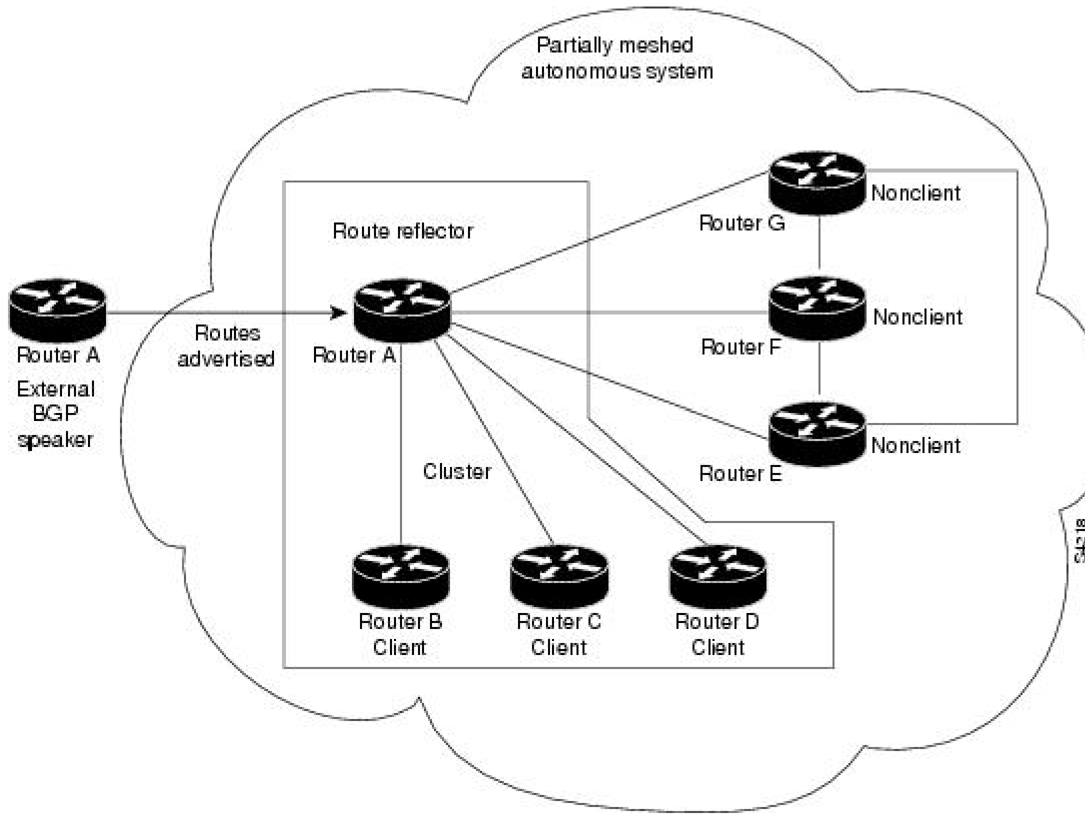
Figure 37: Simple BGP Model with a Route Reflector



The internal peers of the route reflector are divided into two groups: client peers and all the other routers in the autonomous system (nonclient peers). A route reflector reflects routes between these two groups. The route reflector and its client peers form a *cluster*. The nonclient peers must be fully meshed with each other, but the client peers need not be fully meshed. The clients in the cluster do not communicate with iBGP speakers outside their cluster.

The figure below illustrates a more complex route reflector scheme. Router A is the route reflector in a cluster with routers B, C, and D. Routers E, F, and G are fully meshed, nonclient routers.

Figure 38: More Complex BGP Route Reflector Model



When the route reflector receives an advertised route, depending on the neighbor, it takes the following actions:

- A route from an external BGP speaker is advertised to all clients and nonclient peers.
- A route from a nonclient peer is advertised to all clients.
- A route from a client is advertised to all clients and nonclient peers. Hence, the clients need not be fully meshed.

Along with route reflector-aware BGP speakers, it is possible to have BGP speakers that do not understand the concept of route reflectors. They can be members of either client or nonclient groups allowing an easy and gradual migration from the old BGP model to the route reflector model. Initially, you could create a single cluster with a route reflector and a few clients. All the other iBGP speakers could be nonclient peers to the route reflector and then more clusters could be created gradually.

An autonomous system can have multiple route reflectors. A route reflector treats other route reflectors just like other iBGP speakers. A route reflector can be configured to have other route reflectors in a client group or nonclient group. In a simple configuration, the backbone could be divided into many clusters. Each route reflector would be configured with other route reflectors as nonclient peers (thus, all the route reflectors will be fully meshed). The clients are configured to maintain iBGP sessions with only the route reflector in their cluster.

Usually a cluster of clients will have a single route reflector. In that case, the cluster is identified by the router ID of the route reflector. To increase redundancy and avoid a single point of failure, a cluster might have more than one route reflector. In this case, all route reflectors in the cluster must be configured with the 4-byte cluster ID so that a route reflector can recognize updates from route reflectors in the same cluster. All the route reflectors serving a cluster should be fully meshed and all of them should have identical sets of client and nonclient peers.

Route Reflector Mechanisms to Avoid Routing Loops

As the iBGP learned routes are reflected, routing information may loop. The route reflector model has the following mechanisms to avoid routing loops:

- Originator ID is an optional, nontransitive BGP attribute. It is a 4-byte attribute created by a route reflector. The attribute carries the router ID of the originator of the route in the local autonomous system. Therefore, if a misconfiguration causes routing information to come back to the originator, the information is ignored.
- Cluster-list is an optional, nontransitive BGP attribute. It is a sequence of cluster IDs that the route has passed. When a route reflector reflects a route from its clients to nonclient peers, and vice versa, it appends the local cluster ID to the cluster list. If the cluster list is empty, a new cluster list is created. Using this attribute, a route reflector can identify if routing information is looped back to the same cluster due to misconfiguration. If the local cluster ID is found in the cluster list, the advertisement is ignored.
- The use of **set** clauses in outbound route maps can modify attributes and possibly create routing loops. To avoid this behavior, most **set** clauses of outbound route maps are ignored for routes reflected to iBGP peers. The only **set** clause of an outbound route map that is acted upon is the **set ip next-hop** clause.

BGP Outbound Route Map on Route Reflector to Set IP Next Hop for iBGP Peer

The BGP Outbound Route Map on Route Reflector to Set IP Next Hop feature allows a route reflector to modify the next hop attribute for a reflected route.

The use of **set** clauses in outbound route maps can modify attributes and possibly create routing loops. To avoid this behavior, most **set** clauses of outbound route maps are ignored for routes reflected to iBGP peers. The only **set** clause of an outbound route map on a route reflector (RR) that is acted upon is the **set ip next-hop** clause. The **set ip next-hop** clause is applied to reflected routes.

Configuring an RR with an outbound route map allows a network administrator to modify the next hop attribute for a reflected route. By configuring a route map with the **set ip next-hop** clause, the administrator puts the RR into the forwarding path, and can configure iBGP multipath load sharing to achieve load balancing. That is, the RR can distribute outgoing packets among multiple egress points. See the “Configuring iBGP Multipath Load Sharing” module.

Caution Incorrectly setting BGP attributes for reflected routes can cause inconsistent routing, routing loops, or a loss of connectivity. Setting BGP attributes for reflected routes should be attempted only by someone who has a good understanding of the design implications.

BGP Route Dampening

Route dampening is a BGP feature designed to minimize the propagation of flapping routes across an internetwork. A route is considered to be flapping when its availability alternates repeatedly.

For example, consider a network with three BGP autonomous systems: autonomous system 1, autonomous system 2, and autonomous system 3. Suppose the route to network A in autonomous system 1 flaps (it becomes unavailable). Under circumstances without route dampening, the eBGP neighbor of autonomous system 1 to autonomous system 2 sends a withdraw message to autonomous system 2. The border router in autonomous system 2, in turn, propagates the withdraw message to autonomous system 3. When the route to network A reappears, autonomous system 1 sends an advertisement message to autonomous system 2, which sends it to autonomous system 3. If the route to network A repeatedly becomes unavailable, then available, many withdrawal and advertisement messages are sent. This is a problem in an internetwork connected to the Internet because a route flap in the Internet backbone usually involves many routes.

Note No penalty is applied to a BGP peer reset when route dampening is enabled. Although the reset withdraws the route, no penalty is applied in this instance, even if route flap dampening is enabled.

Route Dampening Minimizes Route Flapping

The route dampening feature minimizes the flapping problem as follows. Suppose again that the route to network A flaps. The router in autonomous system 2 (where route dampening is enabled) assigns network A a penalty of 1000 and moves it to history state. The router in autonomous system 2 continues to advertise the status of the route to neighbors. The penalties are cumulative. When the route flaps so often that the penalty exceeds a configurable suppress limit, the router stops advertising the route to network A, regardless of how many times it flaps. Thus, the route is dampened.

The penalty placed on network A is decayed until the reuse limit is reached, upon which the route is once again advertised. At half of the reuse limit, the dampening information for the route to network A is removed.

BGP Route Dampening Terms

The following terms are used when describing route dampening:

- Flap—A route whose availability alternates repeatedly.
- History state—After a route flaps once, it is assigned a penalty and put into history state, meaning the router does not have the best path, based on historical information.
- Penalty—Each time a route flaps, the router configured for route dampening in another autonomous system assigns the route a penalty of 1000. Penalties are cumulative. The penalty for the route is stored in the BGP routing table until the penalty exceeds the suppress limit. At that point, the route state changes from history to damp.
- Damp state—In this state, the route has flapped so often that the router will not advertise this route to BGP neighbors.
- Suppress limit—A route is suppressed when its penalty exceeds this limit. The default value is 2000.
- Half-life—Once the route has been assigned a penalty, the penalty is decreased by half after the half-life period (which is 15 minutes by default). The process of reducing the penalty happens every 5 seconds.
- Reuse limit—As the penalty for a flapping route decreases and falls below this reuse limit, the route is unsuppressed. That is, the route is added back to the BGP table and once again used for forwarding. The default reuse limit is 750. The process of unsuppressing routes occurs at 10-second increments. Every 10 seconds, the router finds out which routes are now unsuppressed and advertises them to the world.
- Maximum suppress limit—This value is the maximum amount of time a route can be suppressed. The default value is four times the half-life.

The routes external to an autonomous system learned via iBGP are not dampened. This policy prevent the iBGP peers from having a higher penalty for routes external to the autonomous system.

BGP Route Map Next Hop Self

The BGP Route Map Next Hop Self feature provides a way to override the settings for bgp next-hop unchanged and bgp next-hop unchanged allpath selectively. These settings are global for an address family. For some routes this may not be appropriate. For example, static routes may need to be redistributed with a next hop of self, but connected routes and routes learned via Interior Border Gateway Protocol (IBGP) or Exterior Border Gateway Protocol (EBGP) may continue to be redistributed with an unchanged next hop.

The BGP route map next hop self functionality modifies the existing route map infrastructure to configure a new ip next-hop self setting, which overrides the bgp next-hop unchanged and bgp next-hop unchanged allpaths settings.

The ip next-hop self setting is applicable only to VPNv4 and VPNv6 address families. Routes distributed by protocols other than BGP are not affected.

You configure a new bgp route-map priority setting to inform BGP that the route map will take priority over the settings for bgp next-hop unchanged and bgp next-hop unchanged allpath. The bgp route-map priority setting only impacts BGP. The bgp route-map priority setting has no impact unless you configure the bgp next-hop unchanged or bgp next-hop unchanged allpaths settings.

How to Configure Internal BGP Features

Configuring a Routing Domain Confederation

To configure a BGP confederation, you must specify a confederation identifier. To the outside world, the group of autonomous systems will look like a single autonomous system with the confederation identifier as the autonomous system number. To configure a BGP confederation identifier, use the following command in router configuration mode:

Command	Purpose
`Router(config-router)#` **`bgp confederation identifier`** *`as-number`*	Configures a BGP confederation.

In order to treat the neighbors from other autonomous systems within the confederation as special eBGP peers, use the following command in router configuration mode:

Command	Purpose
`Router(config-router)#` **`bgp confederation peers`** *`as-number`* `[as-number]`	Specifies the autonomous systems that belong to the confederation.

For an alternative way to reduce the iBGP mesh, see "Configuring a Route Reflector, on page 420."

Configuring a Route Reflector

To configure a route reflector and its clients, use the following command in router configuration mode:

Command	Purpose
Router(config-router)# **neighbor** {*ip-address* | *peer-group-name*} **route-reflector-client**	Configures the local router as a BGP route reflector and the specified neighbor as a client.

If the cluster has more than one route reflector, configure the cluster ID by using the following command in router configuration mode:

Command	Purpose
Router(config-router)# **bgp cluster-id** *cluster-id*	Configures the cluster ID.

Use the **show ip bgp** command to display the originator ID and the cluster-list attributes.

By default, the clients of a route reflector are not required to be fully meshed and the routes from a client are reflected to other clients. However, if the clients are fully meshed, the route reflector need not reflect routes to clients.

To disable client-to-client route reflection, use the **no bgp client-to-client reflection** command in router configuration mode:

Command	Purpose
Router(config-router)# **no bgp client-to-client reflection**	Disables client-to-client route reflection.

Configuring a Route Reflector Using a Route Map to a Set Next Hop for an iBGP Peer

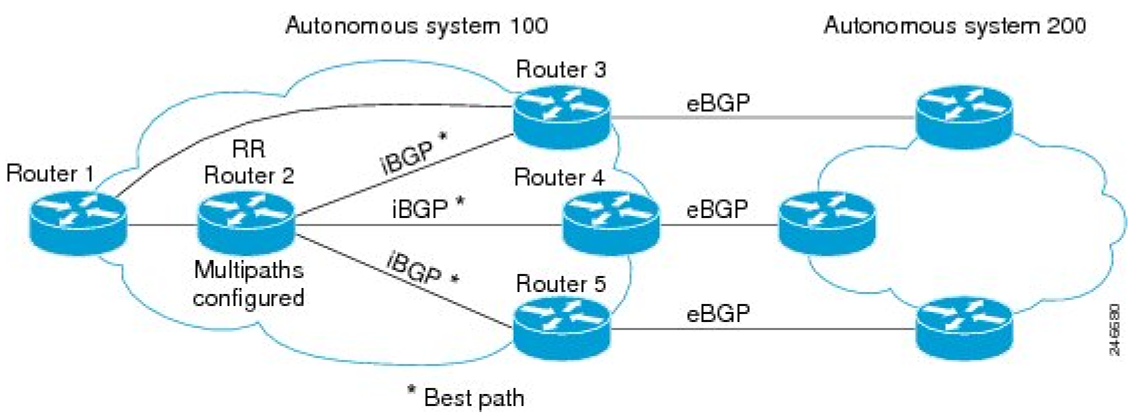
Perform this task on an RR to set a next hop for an iBGP peer. One reason to perform this task is when you want to make the RR the next hop for routes, so that you can configure iBGP load sharing. Create a route map that sets the next hop to be the RR's address, which will be advertised to the RR clients. The route map is applied only to outbound routes from the router to which the route map is applied.

Caution Incorrectly setting BGP attributes for reflected routes can cause inconsistent routing, routing loops, or a loss of connectivity. Setting BGP attributes for reflected routes should only be attempted by someone who has a good understanding of the design implications.

> **Note**
> Do not use the **neighbor next-hop-self** command to modify the next hop attribute for an RR. Using the **neighbor next-hop-self** command on the RR will modify next hop attributes only for non-reflected routes and not the intended routes that are being reflected from the RR clients. To modify the next hop attribute when reflecting a route, use an outbound route map.

This task configures the RR (Router 2) in the scenario illustrated in the figure below. In this case, Router 1 is the iBGP peer whose routes' next hop is being set. Without a route map, outbound routes from Router 1 would go to next hop Router 3. Instead, setting the next hop to the RR's address will cause routes from Router 1 to go to the RR, and thus allow the RR to perform load balancing among Routers 3, 4, and 5.

Figure 39: Route Reflector Using a Route Map to a Set Next Hop for an iBGP Peer



SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag*
4. **set ip next-hop** *ip-address*
5. **exit**
6. **router bgp** *as-number*
7. **address-family ipv4**
8. **maximum-paths ibgp** *number*
9. **neighbor** *ip-address* **remote-as** *as-number*
10. **neighbor** *ip-address* **activate**
11. **neighbor** *ip-address* **route-reflector-client**
12. **neighbor** *ip-address* **route-map** *map-name* **out**
13. Repeat Steps 12 through 14 for the other RR clients.
14. **end**
15. **show ip bgp neighbors**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* **Example:** `Router(config)# route-map rr-out`	Enters route map configuration mode to configure a route map. • The route map is created to set the next hop for the route reflector client.
Step 4	**set ip next-hop** *ip-address* **Example:** `Router(config-route-map)# set ip next-hop 10.2.0.1`	Specifies that for routes that are advertised where this route map is applied, the next-hop attribute is set to this IPv4 address. • For this task, we want to set the next hop to be the address of the RR.
Step 5	**exit** **Example:** `Router(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 6	**router bgp** *as-number* **Example:** `Router(config)# router bgp 100`	Enters router configuration mode and creates a BGP routing process.
Step 7	**address-family ipv4** **Example:** `Router(config-router-af)# address-family ipv4`	Enters address family configuration mode to configure BGP peers to accept address family specific configurations.
Step 8	**maximum-paths ibgp** *number* **Example:** `Router(config-router)# maximum-paths ibgp 5`	Controls the maximum number of parallel iBGP routes that can be installed in the routing table.
Step 9	**neighbor** *ip-address* **remote-as** *as-number* **Example:**	Adds an entry to the BGP neighbor table.

	Command or Action	Purpose
	`Router(config-router-af)# neighbor 10.1.0.1 remote-as 100`	
Step 10	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af)# neighbor 10.1.0.1 activate`	Enables the exchange of information with the peer.
Step 11	**neighbor** *ip-address* **route-reflector-client** **Example:** `Router(config-router-af)# neighbor 10.1.0.1 route-reflector-client`	Configures the local router as a BGP route reflector, and configures the specified neighbor as a route-reflector client.
Step 12	**neighbor** *ip-address* **route-map** *map-name* **out** **Example:** `Router(config-router-af)# neighbor 10.1.0.1 route-map rr-out out`	Applies the route map to outgoing routes from this neighbor. • Reference the route map you created in Step 3.
Step 13	Repeat Steps 12 through 14 for the other RR clients.	You will not be applying a route map to the other RR clients.
Step 14	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.
Step 15	**show ip bgp neighbors** **Example:** `Router# show ip bgp neighbors`	(Optional) Displays information about the BGP neighbors, including their status as RR clients, and information about the route map configured.

Adjusting BGP Timers

BGP uses certain timers to control periodic activities such as the sending of keepalive messages and the interval after not receiving a keepalive message after which the Cisco software declares a peer dead. By default, the keepalive timer is 60 seconds, and the hold-time timer is 180 seconds. You can adjust these timers. When a connection is started, BGP will negotiate the hold time with the neighbor. The smaller of the two hold times will be chosen. The keepalive timer is then set based on the negotiated hold time and the configured keepalive time.

To adjust BGP timers for all neighbors, use the following command in router configuration mode:

Command	Purpose
`Device(config-router)#` **`timers bgp`** *`keepalive holdtime`*	Adjusts BGP timers for all neighbors.

To adjust BGP keepalive and hold-time timers for a specific neighbor, use the following command in router configuration mode:

Command	Purpose
`Device(config-router)#` **neighbor** [*ip-address* | *peer-group-name*] **timers** *keepalive holdtime*	Sets the keepalive and hold-time timers (in seconds) for the specified peer or peer group.

Note The timers configured for a specific neighbor or peer group override the timers configured for all BGP neighbors using the **timers bgp** router configuration command.

To clear the timers for a BGP neighbor or peer group, use the **no** form of the **neighbor timers** command.

Configuring the Router to Consider a Missing MED as the Worst Path

To configure the router to consider a path with a missing MED attribute as the worst path, use the following command in router configuration mode:

Command	Purpose
`Router(config-router)#` **bgp bestpath med missing-as-worst**	Configures the router to consider a missing MED as having a value of infinity, making the path without a MED value the least desirable path.

Configuring the Router to Consider the MED to Choose a Path from Subautonomous System Paths

To configure the router to consider the MED value in choosing a path, use the following command in router configuration mode:

Command	Purpose
`Router(config-router)#` **bgp bestpath med confed**	Configures the router to consider the MED in choosing a path from among those advertised by different subautonomous systems within a confederation.

The comparison between MEDs is made only if there are no external autonomous systems in the path (an external autonomous system is an autonomous system that is not within the confederation). If there is an external autonomous system in the path, then the external MED is passed transparently through the confederation, and the comparison is not made.

The following example compares route A with these paths:

```
path= 65000 65004, med=2
path= 65001 65004, med=3
path= 65002 65004, med=4
path= 65003 1, med=1
```

In this case, path 1 would be chosen if the **bgp bestpath med confed router configuration** command is enabled. The fourth path has a lower MED, but it is not involved in the MED comparison because there is an external autonomous system is in this path.

Configuring the Router to Use the MED to Choose a Path in a Confederation

To configure the router to use the MED to choose the best path from among paths advertised by a single subautonomous system within a confederation, use the following command in router configuration mode:

Command	Purpose
`Router(config-router)# ` **`bgp deterministic med`**	Configures the router to compare the MED variable when choosing among routes advertised by different peers in the same autonomous system.

Note If the **bgp always-compare-med** router configuration command is enabled, all paths are fully comparable, including those from other autonomous systems in the confederation, even if the **bgp deterministic med** command is also enabled.

Configuring BGP Route Dampening

The tasks in this section configure and monitor BGP route dampening. Route dampening is designed to minimize the propagation of flapping routes across an internetwork. A route is considered to be flapping when its availability alternates repeatedly.

Enabling and Configuring BGP Route Dampening

Perform this task to enable and configure BGP route dampening.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** as-*number*
4. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
5. **bgp dampening** [*half-life reuse suppress max-suppress-time*] [**route-map** *map-name*]
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** as-*number* **Example:** `Router(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:** `Router(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 5	**bgp dampening** [*half-life reuse suppress max-suppress-time*] [**route-map** *map-name*] **Example:** `Router(config-router-af)# bgp dampening 30 1500 10000 120`	Enables BGP route dampening and changes the default values of route dampening factors. • The *half-life*, *reuse*, *suppress*, and *max-suppress-time* arguments are all position dependent; if one argument is entered then all the arguments must be entered. • Use the **route-map** keyword and *map-name* argument to control where BGP route dampening is enabled.
Step 6	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Monitoring and Maintaining BGP Route Dampening

Perform the steps in this task as required to monitor and maintain BGP route dampening.

SUMMARY STEPS

1. **enable**
2. **show ip bgp dampening flap-statistics** [**regexp** *regexp* | **filter-list** *access-list* | *ip-address mask* [**longer-prefix**]]

3. **clear ip bgp flap-statistics** [*neighbor-address* [*ipv4-mask*]] [**regexp** *regexp* | **filter-list** *extcom-number*]
4. **show ip bgp dampening dampened-paths**
5. **clear ip bgp** [**ipv4** {**multicast** | **unicast**} | **ipv6** {**multicast** | **unicast**} | **vpnv4 unicast**] **dampening** [*neighbor-address*] [*ipv4-mask*]

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 **show ip bgp dampening flap-statistics** [**regexp** *regexp* | **filter-list** *access-list* | *ip-address mask* [**longer-prefix**]]

Use this command to monitor the flaps of all the paths that are flapping. The statistics will be deleted once the route is not suppressed and is stable for at least one half-life.

Example:

```
Router# show ip bgp dampening flap-statistics

BGP table version is 10, local router ID is 172.17.232.182
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          From            Flaps Duration Reuse    Path
*d 10.0.0.0         172.17.232.177  4     00:13:31 00:18:10 100
*d 10.2.0.0         172.17.232.177  4     00:02:45 00:28:20 100
```

Step 3 **clear ip bgp flap-statistics** [*neighbor-address* [*ipv4-mask*]] [**regexp** *regexp* | **filter-list** *extcom-number*]

Use this command to clear the accumulated penalty for routes that are received on a router that has BGP dampening enabled. If no arguments or keywords are specified, flap statistics are cleared for all routes. Flap statistics are also cleared when the peer is stable for the half-life time period. After the BGP flap statistics are cleared, the route is less likely to be dampened.

Example:

```
Router# clear ip bgp flap-statistics 172.17.232.177
```

Step 4 **show ip bgp dampening dampened-paths**

Use this command to monitor the flaps of all the paths that are flapping. The statistics will be deleted once the route is not suppressed and is stable for at least one half-life.

Example:

```
Router# show ip bgp dampening dampened-paths

BGP table version is 10, local router ID is 172.29.232.182
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          From              Reuse   Path
*d 10.0.0.0         172.16.232.177    00:18:4 100 ?
*d 10.2.0.0         172.16.232.177    00:28:5 100 ?
```

Step 5 **clear ip bgp** [**ipv4** {**multicast** | **unicast**} | **ipv6** {**multicast** | **unicast**} | **vpnv4 unicast**] **dampening** [*neighbor-address*] [*ipv4-mask*]

Use this command to clear stored route dampening information. If no keywords or arguments are entered, route dampening information for the entire routing table is cleared. The following example clears route dampening information for VPNv4 address family prefixes from network 192.168.10.0/24, and unsuppresses its suppressed routes.

Example:

```
Router# clear ip bgp vpnv4 unicast dampening 192.168.10.0 255.255.255.0
```

Monitoring and Maintaining BGP Route Dampening

You can monitor the flaps of all the paths that are flapping. The statistics will be deleted once the route is not suppressed and is stable for at least one half-life. To display flap statistics, use the following commands as needed:

Command	Purpose
`Router# show ip bgp dampening flap-statistics`	Displays BGP flap statistics for all paths.
`Router# show ip bgp dampening flap-statistics regexp regexp`	Displays BGP flap statistics for all paths that match the regular expression.
`Router# show ip bgp dampening flap-statistics filter-list access- list`	Displays BGP flap statistics for all paths that pass the filter.
`Router# show ip bgp dampening flap-statistics ip-address mask`	Displays BGP flap statistics for a single entry.
`Router# show ip bgp dampening flap-statistics ip-address mask longer-prefix`	Displays BGP flap statistics for more specific entries.

To clear BGP flap statistics (thus making it less likely that the route will be dampened), use the following commands as needed:

Command	Purpose
`Router# clear ip bgp flap-statistics`	Clears BGP flap statistics for all routes.
`Router# clear ip bgp flap-statistics regexp regexp`	Clears BGP flap statistics for all paths that match the regular expression.
`Router# clear ip bgp flap-statistics filter-list list`	Clears BGP flap statistics for all paths that pass the filter.

Command	Purpose
`Router# clear ip bgp flap-statistics ip-address mask`	Clears BGP flap statistics for a single entry.
`Router# clear ip bgp ip-address flap-statistics`	Clears BGP flap statistics for all paths from a neighbor.

Note The flap statistics for a route are also cleared when a BGP peer is reset. Although the reset withdraws the route, there is no penalty applied in this instance, even if route flap dampening is enabled.

Once a route is dampened, you can display BGP route dampening information, including the time remaining before the dampened routes will be unsuppressed. To display the information, use the following command:

Command	Purpose
`Router# show ip bgp dampening dampened-paths`	Displays the dampened routes, including the time remaining before they will be unsuppressed.

You can clear BGP route dampening information and unsuppress any suppressed routes by using the following command:

Command	Purpose
`Router# clear ip bgp dampened-paths [ip-address network-mask]`	Clears route dampening information and unsuppresses the suppressed routes.

Configuring BGP Route Map next-hop self

Perform this task to modify the existing route map by adding the ip next-hop self setting and overriding the bgp next-hop unchanged and bgp next-hop unchanged allpaths settings.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **route-map** *map-tag* **permit** *sequence-number*
4. **match source-protocol** *source-protocol*
5. **set ip next-hop self**
6. **exit**
7. **route-map** *map-tag* **permit** *sequence-number*
8. **match route-type internal**
9. **match route-type external**
10. **match source-protocol** *source-protocol*

11. **exit**
12. **router bgp** *autonomous-system-number*
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **address-family vpnv4**
15. **neighbor** *ip-address* **activate**
16. **neighbor** *ip-address* **next-hop unchanged allpaths**
17. **neighbor** *ip-address* **route-map** *map-name* **out**
18. **exit**
19. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
20. **bgp route-map priority**
21. **redistribute** *protocol*
22. **redistribute** *protocol*
23. **exit-address-family**
24. **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	**route-map** *map-tag* **permit** *sequence-number* **Example:** `Device(config)# route-map static-nexthop-rewrite permit 10`	Defines conditions for redistributing routes from one routing protocol to another routing protocol and enters route-map configuration mode.
Step 4	**match source-protocol** *source-protocol* **Example:** `Device(config-route-map)# match source-protocol static`	Matches Enhanced Interior Gateway Routing Protocol (EIGRP) external routes based on a source protocol.
Step 5	**set ip next-hop self** **Example:** `Device(config-route-map)# set ip next-hop self`	Configure local routes (for BGP only) with next hop of self.
Step 6	**exit** **Example:**	Exits route-map configuration mode and enters global configuration mode.

	Command or Action	Purpose
	`Device(config-route-map)# exit`	
Step 7	**route-map** *map-tag* **permit** *sequence-number* **Example:** `Device(config)# route-map static-nexthop-rewrite permit 20`	Defines conditions for redistributing routes from one routing protocol to another routing protocol and enters route-map configuration mode.
Step 8	**match route-type internal** **Example:** `Device(config-route-map)# match route-type internal`	Redistributes routes of the specified type.
Step 9	**match route-type external** **Example:** `Device(config-route-map)# match route-type external`	Redistributes routes of the specified type.
Step 10	**match source-protocol** *source-protocol* **Example:** `Device(config-route-map)# match source-protocol connected`	Matches Enhanced Interior Gateway Routing Protocol (EIGRP) external routes based on a source protocol.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 12	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 172.16.232.50 remote-as 65001`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 14	**address-family vpnv4** **Example:** `Device(config-router)# address-family vpnv4`	Specifies the VPNv4 address family and enters address family configuration mode.

	Command or Action	Purpose
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 172.16.232.50 activate`	Enables the exchange of information with a Border Gateway Protocol (BGP) neighbor.
Step 16	**neighbor** *ip-address* **next-hop unchanged allpaths** **Example:** `Device(config-router-af)# neighbor 172.16.232.50 next-hop unchanged allpaths`	Enables an external EBGP peer that is configured as multihop to propagate the next hop unchanged.
Step 17	**neighbor** *ip-address* **route-map** *map-name* **out** **Example:** `Device(config-router-af)# neighbor 172.16.232.50 route-map static-nexthop-rewrite out`	Applies a route map to an outgoing route.
Step 18	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 19	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast vrf inside`	Specifies the IPv4 address family and enters address family configuration mode.
Step 20	**bgp route-map priority** **Example:** `Device(config-router-af)# bgp route-map priority`	Configures the route map priority for the local BGP routing process
Step 21	**redistribute** *protocol* **Example:** `Device(config-router-af)# redistribute static`	Redistributes routes from one routing domain into another routing domain.
Step 22	**redistribute** *protocol* **Example:** `Device(config-router-af)# redistribute connected`	Redistributes routes from one routing domain into another routing domain.
Step 23	**exit-address-family** **Example:** `Device(config-router-af)# exit address-family`	Exits address family configuration mode and enters router configuration mode .

	Command or Action	Purpose
Step 24	**end** **Example:** `Device(config-router)# end`	Exits router configuration mode and enters privileged EXEC mode.

Configuration Examples for Internal BGP Features

Example: BGP Confederation Configurations with Route Maps

This section contains an example of the use of a BGP confederation configuration that includes BGP communities and route maps. For more examples of how to configure a BGP confederation, see the "Example: BGP Confederation" section in this module

This example shows how BGP community attributes are used with a BGP confederation configuration to filter routes.

In this example, the route map named *set-community* is applied to the outbound updates to neighbor 172.16.232.50 and the local-as community attribute is used to filter the routes. The routes that pass access list 1 have the special community attribute value local-as. The remaining routes are advertised normally. This special community value automatically prevents the advertisement of those routes by the BGP speakers outside autonomous system 200.

```
router bgp 65000
 network 10.0.1.0 route-map set-community
 bgp confederation identifier 200
 bgp confederation peers 65001
 neighbor 172.16.232.50 remote-as 100
 neighbor 172.16.233.2 remote-as 65001
!
route-map set-community permit 10
 match ip address 1
 set community local-as
!
```

Example: BGP Confederation

The following is a sample configuration that shows several peers in a confederation. The confederation consists of three internal autonomous systems with autonomous system numbers 6001, 6002, and 6003. To the BGP speakers outside the confederation, the confederation looks like a normal autonomous system with autonomous system number 500 (specified via the **bgp confederation identifier** router configuration command).

In a BGP speaker in autonomous system 6001, the **bgp confederation peers** router configuration command marks the peers from autonomous systems 6002 and 6003 as special eBGP peers. Hence peers 172.16.232.55 and 172.16.232.56 will get the local preference, next hop, and MED unmodified in the updates. The router at 10.16.69.1 is a normal eBGP speaker and the updates received by it from this peer will be just like a normal eBGP update from a peer in autonomous system 6001.

```
router bgp 6001
 bgp confederation identifier 500
```

```
 bgp confederation peers 6002 6003
 neighbor 172.16.232.55 remote-as 6002
 neighbor 172.16.232.56 remote-as 6003
 neighbor 10.16.69.1 remote-as 777
```

In a BGP speaker in autonomous system 6002, the peers from autonomous systems 6001 and 6003 are configured as special eBGP peers. 10.70.70.1 is a normal iBGP peer and 10.99.99.2 is a normal eBGP peer from autonomous system 700.

```
router bgp 6002
 bgp confederation identifier 500
 bgp confederation peers 6001 6003
 neighbor 10.70.70.1 remote-as 6002
 neighbor 172.16.232.57 remote-as 6001
 neighbor 172.16.232.56 remote-as 6003
 neighbor 10.99.99.2 remote-as 700
```

In a BGP speaker in autonomous system 6003, the peers from autonomous systems 6001 and 6002 are configured as special eBGP peers. 10.200.200.200 is a normal eBGP peer from autonomous system 701.

```
router bgp 6003
 bgp confederation identifier 500
 bgp confederation peers 6001 6002
 neighbor 172.16.232.57 remote-as 6001
 neighbor 172.16.232.55 remote-as 6002
 neighbor 10.200.200.200 remote-as 701
```

The following is a part of the configuration from the BGP speaker 10.200.200.205 from autonomous system 701 in the same example. Neighbor 172.16.232.56 is configured as a normal eBGP speaker from autonomous system 500. The internal division of the autonomous system into multiple autonomous systems is not known to the peers external to the confederation.

```
router bgp 701
 neighbor 172.16.232.56 remote-as 500
 neighbor 10.200.200.205 remote-as 701
```

Example: Route Reflector Using a Route Map to Set a Next Hop for an iBGP Peer

The following example is based on the figure above. Router 2 is the route reflector for the clients: Routers 1, 3, 4, and 5. Router 1 is connected to Router 3, but you don’t want Router 1 to forward traffic destined to AS 200 to use Router 3 as the next hop (and therefore use the direct link with Router 3); you want to direct the traffic to the RR, which can load share among Routers 3, 4, and 5.

This example configures the RR, Router 2. A route map named rr-out is applied to Router 1; the route map sets the next hop to be the RR at 10.2.0.1. When Router 1 sees that the next hop is the RR address, Router 1 forwards the routes to the RR. When the RR receives packets, it will automatically load share among the iBGP paths. A maximum of five iBGP paths are allowed.

Router 2

```
route-map rr-out
 set ip next-hop 10.2.0.1
!
interface gigabitethernet 0/0
 ip address 10.2.0.1 255.255.0.0
```

```
router bgp 100
 address-family ipv4 unicast
 maximum-paths ibgp 5
 neighbor 10.1.0.1 remote-as 100
 neighbor 10.1.0.1 activate
 neighbor 10.1.0.1 route-reflector-client
 neighbor 10.1.0.1 route-map rr-out out
!
 neighbor 10.3.0.1 remote-as 100
 neighbor 10.3.0.1 activate
 neighbor 10.3.0.1 route-reflector-client
!
 neighbor 10.4.0.1 remote-as 100
 neighbor 10.4.0.1 activate
 neighbor 10.4.0.1 route-reflector-client
!
 neighbor 10.5.0.1 remote-as 100
 neighbor 10.5.0.1 activate
 neighbor 10.5.0.1 route-reflector-client
end
```

Example: Configuring BGP Route Dampening

The following example configures BGP dampening to be applied to prefixes filtered through the route map named ACCOUNTING:

```
ip prefix-list FINANCE permit 10.0.0.0/8
!
route-map ACCOUNTING
 match ip address ip prefix-list FINANCE
 set dampening 15 750 2000 60
 exit
router bgp 50000
 address-family ipv4
 bgp dampening route-map ACCOUNTING
 end
```

Example: Configuring BGP Route Map next-hop self

This section contains an example of how to configure BGP Route Map next-hop self.

In this example, a route map is configured that matches the networks where you wish to override settings for bgp next-hop unchanged and bgp next-hop unchanged allpath. Subsequently, next-hop self is configured. After this, the bgp route map priority is configured for the specified address family so that the previously specified route map takes priority over the settings for bgp next-hop unchanged and bgp next-hop unchanged allpath. This configuration results in static routes being redistributed with a next hop of self, but connected routes and routes learned via IBGP or EBGP continue to be redistributed with an unchanged next hop.

```
route-map static-nexthop-rewrite permit 10
 match source-protocol static
 set ip next-hop self
route-map static-nexthop-rewrite permit 20
 match route-type internal
 match route-type external
 match source-protocol connected
!
router bgp 65000
```

```
 neighbor 172.16.232.50 remote-as 65001
 address-family vpnv4
   neighbor 172.16.232.50 activate
   neighbor 172.16.232.50 next-hop unchanged allpaths
   neighbor 172.16.232.50 route-map static-nexthop-rewrite out
 exit-address-family
 address-family ipv4 unicast vrf inside
   bgp route-map priority
   redistribute static
   redistribute connected
 exit-address-family
end
```

Additional References for Internal BGP Features

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
BGP overview	“Cisco BGP Overview” module
Basic BGP configuration tasks	“Configuring a Basic BGP Network” module
iBGP multipath load sharing	“iBGP Multipath Load Sharing” module
Connecting to a service provider	“Connecting to a Service Provider Using External BGP” module
Configuring features that apply to multiple IP routing protocols	*IP Routing: Protocol-Independent Configuration Guide*

RFCs

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 of 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	Title
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5396	*Textual Representation of Autonomous system (AS) Numbers*
RFC 5398	*Autonomous System (AS) Number Reservation for Documentation Use*

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 Configuring Internal BGP Features

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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 31: Feature Information for Configuring Internal BGP Features

Feature Name	Releases	Feature Configuration Information
Configuring internal BGP features	10.3 12.0(7)T 12.0(32)S12 12.2(33)SRA 12.2(33)SXH	All the features contained in this module are considered to be legacy features and will work in all trains release images. The following commands were introduced or modified by these features: • **bgp always-compare-med** • **bgp bestpath med confed** • **bgp bestpath med missing-as-worst** • **bgp client-to-client reflection** • **bgp cluster-id** • **bgp confederation identifier** • **bgp confederation peers** • **bgp dampening** • **bgp deterministic med** • **clear ip bgp dampening** • **clear ip bgp flap-statistics** • **neighbor route-reflector-client** • **neighbor timers** • **show ip bgp** • **show ip bgp dampening dampened-paths** • **show ip bgp dampening flap-statistics** • **timers bgp**
BGP Outbound Route Map on Route Reflector to Set IP Next Hop	12.0(16)ST 12.0(22)S 12.2 12.2(14)S 15.0(1)S	The BGP Outbound Route Map on Route Reflector to Set IP Next Hop feature allows a route reflector to modify the next hop attribute for a reflected route.

CHAPTER 19

BGP Support for Next-Hop Address Tracking

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

Finding Feature Information

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

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

Information About BGP Support for Next-Hop Address Tracking

BGP Next-Hop Address Tracking

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

BGP Next-Hop Dampening Penalties

If the penalty threshold value is higher than 950, then the delay is calculated as the reuse time using the dampening calculations. The dampening calculations use the following parameters:

- Penalty
- Half-life time
- Reuse time
- max-suppress-time

The values for the dampening parameters used are a max-suppress-time of 60 seconds, the half-life of 8 seconds, and the reuse-limit of 100.

For example, if the original penalty of 1600 is added, then after 16 seconds it becomes 800, and after 40 seconds, the penalty becomes 100. Hence, for the route update penalty of 1600, a delay of 40 seconds is used to schedule the BGP scanner.

These parameters (penalty threshold and any of the dampening parameters) cannot be modified.

Default BGP Scanner Behavior

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

BGP Next_Hop Attribute

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

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

Selective BGP Next-Hop Route Filtering

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

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

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

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

BGP Support for Fast Peering Session Deactivation

BGP Hold Timer

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

BGP Fast Peering Session Deactivation

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

Selective Address Tracking for BGP Fast Session Deactivation

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

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

How to Configure BGP Support for Next-Hop Address Tracking

Configuring BGP Next-Hop Address Tracking

The tasks in this section show how configure BGP next-hop address tracking. BGP next-hop address tracking significantly improves the response time of BGP to next-hop changes in the RIB. However, unstable Interior

Gateway Protocol (IGP) peers can introduce instability to BGP neighbor sessions. We recommend that you aggressively dampen unstable IGP peering sessions to reduce the possible impact to BGP. For more details about configuring route dampening, see "Configuring BGP Route Dampening."

Configuring BGP Selective Next-Hop Route Filtering

Perform this task to configure selective next-hop route filtering using a route map to filter potential next-hop routes. This task uses prefix lists and route maps to match IP addresses or source protocols and can be used to avoid aggregate addresses and BGP prefixes being considered as next-hop routes. Only **match ip address** and **match source-protocol** commands are supported in the route map. No **set** commands or other **match** commands are supported.

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:**	Enters router configuration mode and creates a BGP routing process.

	Command or Action	Purpose
	`Device(config)# router bgp 45000`	
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 with the **address-family ipv4** command. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**bgp nexthop route-map** *map-name* **Example:** `Device(config-router-af)# bgp nexthop route-map CHECK-NEXTHOP`	Permits a route map to selectively define routes to help resolve the BGP next hop. • In this example the route map named CHECK-NEXTHOP is created.
Step 6	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 7	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 8	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network* / *length* | **permit** *network*/*length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list FILTER25 seq 5 permit 0.0.0.0/0 le 25`	Creates a prefix list for BGP next-hop route filtering. • Selective next-hop route filtering supports prefix length matching or source protocol matching on a per address-family basis. • The example creates a prefix list named FILTER25 that permits routes only if the mask length is more than 25; this will avoid aggregate routes being considered as the next-hop route.
Step 9	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:**	Configures a route map and enters route map configuration mode. • In this example, a route map named CHECK-NEXTHOP is created. If there is an IP

	Command or Action	Purpose
	`Device(config)# route-map CHECK-NEXTHOP deny 10`	address match in the following **match** command, the IP address will be denied.
Step 10	**match ip address prefix-list** *prefix-list-name* [*prefix-list-name...*] **Example:** `Device(config-route-map)# match ip address prefix-list FILTER25`	Matches the IP addresses in the specified prefix list. • Use the *prefix-list-name* argument to specify the name of a prefix list. The ellipsis means that more than one prefix list can be specified. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route map configuration mode and enters global configuration mode.
Step 12	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map CHECK-NEXTHOP permit 20`	Configures a route map and enters route map configuration mode. • In this example, all other IP addresses are permitted by route map CHECK-NEXTHOP.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.
Step 14	**show ip bgp** [*network*] [*network-mask*] **Example:** `Device# show ip bgp`	Displays the entries in the BGP routing table. • Enter this command to view the next-hop addresses for each route. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

Example

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

```
BGP table version is 7, local router ID is 172.17.1.99
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*  10.1.1.0/24      192.168.1.2              0             0 40000 i
*  10.2.2.0/24      192.168.3.2              0             0 50000 i
```

```
*> 172.16.1.0/24    0.0.0.0                  0         32768 i
*> 172.17.1.0/24    0.0.0.0                  0         32768
```

Adjusting the Delay Interval for BGP Next-Hop Address Tracking

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `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 64512`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**address-family ipv4** [[**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]] **Example:** `Device(config-router)# address-family ipv4 unicast`	Enter address family configuration mode to configure BGP peers to accept address family-specific configurations. • The example creates an IPv4 unicast address family session.

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

Disabling BGP Next-Hop Address Tracking

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

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

SUMMARY STEPS

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

DETAILED STEPS

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

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

Configuring Fast Session Deactivation

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

Configuring Fast Session Deactivation for a BGP Neighbor

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

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*

4. **address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
5. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **fall-over**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `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 to create or configure a BGP routing process.
Step 4	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure BGP peers to accept address family-specific configurations. • The example creates an IPv4 unicast address family session.
Step 5	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 10.0.0.1 remote-as 50000`	Establishes a peering session with a BGP neighbor.
Step 6	**neighbor** *ip-address* **fall-over** **Example:** `Device(config-router-af)# neighbor 10.0.0.1 fall-over`	Configures the BGP peering to use fast session deactivation. • BGP will remove all routes learned through this peer if the session is deactivated.
Step 7	**end** **Example:** `Device(config-router-af)# end`	Exits configuration mode and returns to privileged EXEC mode.

Configuring Selective Address Tracking for Fast Session Deactivation

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `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:**	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.

	Command or Action	Purpose
	`Device(config-router)# neighbor 192.168.1.2 remote-as 40000`	
Step 5	**neighbor** *ip-address* **fall-over** [**route-map** *map-name*] **Example:** `Device(config-router)# neighbor 192.168.1.2 fall-over route-map CHECK-NBR`	Applies a route map when a route to the BGP changes. • In this example, the route map named CHECK-NBR is applied when the route to neighbor 192.168.1.2 changes.
Step 6	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 7	**ip prefix-list** *list-name* [**seq** *seq-value*]{**deny** *network* / *length* | **permit** *network* / *length*}[**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list FILTER28 seq 5 permit 0.0.0.0/0 ge 28`	Creates a prefix list for BGP next-hop route filtering. • Selective next-hop route filtering supports prefix length matching or source protocol matching on a per-address-family basis. • The example creates a prefix list named FILTER28 that permits routes only if the mask length is greater than or equal to 28.
Step 8	**route-map** *map-name* [**permit** | **deny**][*sequence-number*] **Example:** `Device(config)# route-map CHECK-NBR permit 10`	Configures a route map and enters route-map configuration mode. • In this example, a route map named CHECK-NBR is created. If there is an IP address match in the following **match** command, the IP address will be permitted.
Step 9	**match ip address prefix-list** *prefix-list-name* [*prefix-list-name...*] **Example:** `Device(config-route-map)# match ip address prefix-list FILTER28`	Matches the IP addresses in the specified prefix list. • Use the *prefix-list-name* argument to specify the name of a prefix list. The ellipsis means that more than one prefix list can be specified. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 10	**end** **Example:** `Device(config-route-map)# end`	Exits configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP Support for Next-Hop Address Tracking

Example: Enabling and Disabling BGP Next-Hop Address Tracking

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

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

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

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

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

Examples: Configuring BGP Selective Next-Hop Route Filtering

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

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

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

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

```
 exit
route-map CHECK-BGP25 permit 30
 end
```

Example: Configuring Fast Session Deactivation for a BGP Neighbor

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

Device A

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

Device B

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

Example: Configuring Selective Address Tracking for Fast Session Deactivation

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

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

Additional References

Related Documents

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

Technical Assistance

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

Feature Information for BGP Support for Next-Hop Address Tracking

Table 32: Feature Information for BGP Support for Next-Hop Address Tracking

Feature Name	Releases	Feature Information
BGP Support for Next-Hop Address Tracking	12.3(14)T	The BGP Support for Next-Hop Address Tracking feature is enabled by default when a supporting Cisco IOS software image is installed. BGP next-hop address tracking is event driven. BGP prefixes are automatically tracked as peering sessions are established. Next-hop changes are rapidly reported to the BGP routing process as they are updated in the RIB. This optimization improves overall BGP convergence by reducing the response time to next-hop changes for routes installed in the RIB. When a bestpath calculation is run in between BGP scanner cycles, only next-hop changes are tracked and processed. The following command was introduced in this feature: **bgp nexthop**.

Feature Name	Releases	Feature Information
BGP Selective Address Tracking	12.4(4)T	The BGP Selective Address Tracking feature introduces the use of a route map for next-hop route filtering and fast session deactivation. Selective next-hop filtering uses a route map to selectively define routes to help resolve the BGP next hop, or a route map can be used to determine if a peering session with a BGP neighbor should be reset when a route to the BGP peer changes. The following commands were modified by this feature: **bgp nexthop**, **neighbor fall-over**.
BGP Support for Fast Peering Session Deactivation	12.3(14)T	The BGP Support for Fast Peering Session Deactivation feature introduced an event-driven notification system that allows a Border Gateway Protocol (BGP) process to monitor BGP peering sessions on a per-neighbor basis. This feature improves the response time of BGP to adjacency changes by allowing BGP to detect an adjacency change and deactivate the terminated session in between standard BGP scanning intervals. Enabling this feature improves overall BGP convergence. The following command was modified by this feature: **neighbor fall-over**.

CHAPTER 20

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**	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 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*...] **Example:** `Device(config-router)# neighbor 10.4.9.5 remote-as 100`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 8	**neighbor** {*ip-address* | *ipv6-address***%** | } **maximum-prefix** *maximum* [*threshold*] [**restart** *minutes*] [**warning-only**] **Example:**	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

	Command or Action	Purpose
	`Device(config-router)# neighbor 10.4.9.5 maximum-prefix 1000 90 restart 60`	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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 33: Feature Information for BGP Restart Session After Max-Prefix Limit

Feature Name	Releases	Feature Information
BGP Restart Session After Max-Prefix Limit	12.2(15)T	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. The following commands were modified: **neighbor maximum-prefix** and **show ip bgp neighbors**.
BGP—Subcodes for BGP Cease Notification	15.5(1)T	Support for subcodes for BGP cease notification has been added.

CHAPTER 21

BGP Link Bandwidth

The BGP (Border Gateway Protocol) Link Bandwidth feature is used to advertise the bandwidth of an autonomous system exit link as an extended community. This feature is configured for links between directly connected external BGP (eBGP) neighbors. The link bandwidth extended community attribute is propagated to iBGP peers when extended community exchange is enabled. This feature is used with BGP multipath features to configure load balancing over links with unequal bandwidth.

Finding Feature Information

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

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

Prerequisites for BGP Link Bandwidth

- BGP load balancing or multipath load balancing must be configured before BGP Link Bandwidth feature is enabled.
- BGP extended community exchange must be enabled between iBGP neighbors to which the link bandwidth attribute is to be advertised.
- Cisco Express Forwarding or distributed Cisco Express Forwarding must be enabled on all participating routers.

Restrictions for BGP Link Bandwidth

- The BGP Link Bandwidth feature can be configured only under IPv4 and VPNv4 address family sessions.
- BGP can originate the link bandwidth community only for directly connected links to eBGP neighbors.
- Both iBGP and eBGP load balancing are supported in IPv4 and VPNv4 address families. However, eiBGP load balancing is supported only in VPNv4 address families.

Information About BGP Link Bandwidth

BGP Link Bandwidth Overview

The BGP Link Bandwidth feature is used to enable multipath load balancing for external links with unequal bandwidth capacity. This feature is enabled under an IPv4 or VPNv4 address family session by entering the **bgp dmzlink-bw** command. This feature supports iBGP, eBGP multipath load balancing, and eiBGP multipath load balancing in Multiprotocol Label Switching (MPLS) VPNs. When this feature is enabled, routes learned from directly connected external neighbor are propagated through the internal BGP (iBGP) network with the bandwidth of the source external link.

The link bandwidth extended community indicates the preference of an autonomous system exit link in terms of bandwidth. This extended community is applied to external links between directly connected eBGP peers by entering the **neighbor dmzlink-bw** command. The link bandwidth extended community attribute is propagated to iBGP peers when extended community exchange is enabled with the **neighbor send-community** command.

Link Bandwidth Extended Community Attribute

The link bandwidth extended community attribute is a 4-byte value that is configured for a link on the demilitarized zone (DMZ) interface that connects two single hop eBGP peers. The link bandwidth extended community attribute is used as a traffic sharing value relative to other paths while traffic is being forwarded. Two paths are designated as equal for load balancing if the weight, local-pref, as-path length, Multi Exit Discriminator (MED), and Interior Gateway Protocol (IGP) costs are the same.

Benefits of the BGP Link Bandwidth Feature

The BGP Link Bandwidth feature allows BGP to be configured to send traffic over multiple iBGP or eBGP learned paths where the traffic that is sent is proportional to the bandwidth of the links that are used to exit the autonomous system. The configuration of this feature can be used with eBGP and iBGP multipath features to enable unequal cost load balancing over multiple links. Unequal cost load balancing over links with unequal bandwidth was not possible in BGP before the BGP Link Bandwidth feature was introduced.

How to Configure BGP Link Bandwidth

Configuring and Verifying BGP Link Bandwidth

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4**
5. **address-family ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*]
6. **bgp dmzlink-bw**
7. **neighbor** *ip-address* **dmzlink-bw**
8. **neighbor** *ip-address* **send-community** [**both** | **extended** | **standard**]
9. **end**
10. **show ip bgp** *ip-address* [**longer-prefixes** [**injected**] | **shorter-prefixes** [*mask-length*]]
11. **show ip route** *ip-address* [*mask*] [**longer-prefixes**] | *protocol* [*process-id*] | [**list** *access-list-number* | *access-list-name*] | **static download**]

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 to create or configure a BGP routing process.
Step 4	**address-family ipv4** **Example:** `Device(config-router-af)# address-family ipv4`	Enters address family configuration mode.
Step 5	**address-family ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:**	The BGP Link Bandwidth feature is supported only under the IPv4 and VPNv4 address families.

	Command or Action	Purpose
	`Device(config-router)# address-family ipv4`	
Step 6	**bgp dmzlink-bw** **Example:** `Router(config-router-af)# bgp dmzlink-bw`	Configures BGP to distribute traffic proportionally to the bandwidth of the link. • This command must be entered on each router that contains an external interface that is to be used for multipath load balancing.
Step 7	**neighbor** *ip-address* **dmzlink-bw** **Example:** `Device(config-router-af)# neighbor 172.16.1.1 dmzlink-bw`	Configures BGP to include the link bandwidth attribute for routes learned from the external interface specified IP address. • This command must be configured for each eBGP link that is to be configured as a multipath. Enabling this command allows the bandwidth of the external link to be propagated through the link bandwidth extended community.
Step 8	**neighbor** *ip-address* **send-community** [**both** | **extended** | **standard**] **Example:** `Device(config-router-af)# neighbor 10.10.10.1 send-community extended`	(Optional) Enables community or extended community exchange with the specified neighbor. • This command must be configured for iBGP peers to which the link bandwidth extended community attribute is to be propagated.
Step 9	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode, and enters privileged EXEC mode.
Step 10	**show ip bgp** *ip-address* [**longer-prefixes** [**injected**] | **shorter-prefixes** [*mask-length*]] **Example:** `Device# show ip bgp 10.0.0.0`	(Optional) Displays information about the TCP and BGP connections to neighbors. • The output displays the status of the link bandwidth configuration. The bandwidth of the link is shown in kilobytes.
Step 11	**show ip route** *ip-address* [*mask*] [**longer-prefixes**] | *protocol* [*process-id*] | [**list** *access-list-number* | *access-list-name*] | **static download**] **Example:** `Device# show ip route 10.0.0.0`	(Optional) Displays the current state of the routing table. • The output displays traffic share values, including the weights of the links that are used to direct traffic proportionally to the bandwidth of each link.

Configuration Examples for BGP Link Bandwidth

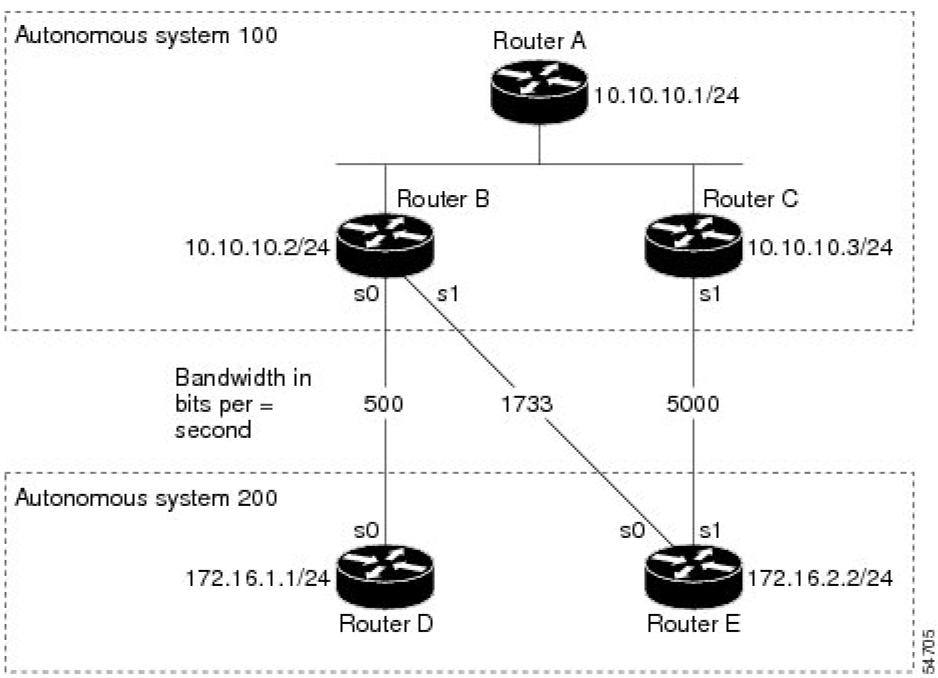
Example: BGP Link Bandwidth Configuration

In the following examples, the BGP Link Bandwidth feature is configured so BGP will distribute traffic proportionally to the bandwidth of each external link. The figure below shows two external autonomous systems connected by three links that each carry a different amount of bandwidth (unequal cost links). Multipath load balancing is enabled and traffic is balanced proportionally.

Note The BGP Link Bandwidth feature functions for simple topologies that have a single path toward the exit points.

Caution The BGP Link Bandwidth feature might not function properly if load balancing is required toward the exit points.

Figure 40: BGP Link Bandwidth Configuration



Router A Configuration

In the following example, Router A is configured to support iBGP multipath load balancing and to exchange the BGP extended community attribute with iBGP neighbors:

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

```
RouterA(config-router)# neighbor 10.10.10.2 remote-as 100

RouterA(config-router)# neighbor 10.10.10.2 update-source Loopback 0

RouterA(config-router)# neighbor 10.10.10.3 remote-as 100

RouterA(config-router)# neighbor 10.10.10.3 update-source Loopback 0

RouterA(config-router)# address-family ipv4

RouterA(config-router-af)# bgp dmzlink-bw
RouterA(config-router-af)# neighbor 10.10.10.2 activate

RouterA(config-router-af)# neighbor 10.10.10.2 send-community both

RouterA(config-router-af)# neighbor 10.10.10.3 activate

RouterA(config-router-af)# neighbor 10.10.10.3 send-community both

RouterA(config-router-af)# maximum-paths ibgp 6
```

Router B Configuration

In the following example Router B is configured to support multipath load balancing, to distribute Router D and Router E link traffic proportionally to the bandwidth of each link, and to advertise the bandwidth of these links to iBGP neighbors as an extended community:

```
RouterB(config)# router bgp 100

RouterB(config-router)# neighbor 10.10.10.1 remote-as 100

RouterB(config-router)# neighbor 10.10.10.1 update-source Loopback 0

RouterB(config-router)# neighbor 10.10.10.3 remote-as 100

RouterB(config-router)# neighbor 10.10.10.3 update-source Loopback 0
RouterB(config-router)# neighbor 172.16.1.1 remote-as 200

RouterB(config-router)# neighbor 172.16.1.1 ebgp-multihop 1
RouterB(config-router)# neighbor 172.16.2.2 remote-as 200

RouterB(config-router)# neighbor 172.16.2.2 ebgp-multihop 1
RouterB(config-router)# address-family ipv4

RouterB(config-router-af)# bgp dmzlink-bw

RouterB(config-router-af)# neighbor 10.10.10.1 activate

RouterB(config-router-af)# neighbor 10.10.10.1 next-hop-self

RouterB(config-router-af)# neighbor 10.10.10.1 send-community both

RouterB(config-router-af)# neighbor 10.10.10.3 activate

RouterB(config-router-af)# neighbor 10.10.10.3 next-hop-self

RouterB(config-router-af)# neighbor 10.10.10.3 send-community both

RouterB(config-router-af)# neighbor 172.16.1.1
```

```
 activate
RouterB(config-router-af)# neighbor 172.16.1.1 dmzlink-bw

RouterB(config-router-af)# neighbor 172.16.2.2 activate
RouterB(config-router-af)# neighbor 172.16.2.2 dmzlink-bw
RouterB(config-router-af)# maximum-paths ibgp 6
RouterB(config-router-af)# maximum-paths 6
```

Router C Configuration

In the following example Router C is configured to support multipath load balancing and to advertise the bandwidth of the link with Router E to iBGP neighbors as an extended community:

```
RouterC(config)# router bgp 100
RouterC(config-router)# neighbor 10.10.10.1 remote-as 100
RouterC(config-router)# neighbor 10.10.10.1 update-source Loopback 0
RouterC(config-router)# neighbor 10.10.10.2 remote-as 100
RouterC(config-router)# neighbor 10.10.10.2 update-source Loopback 0
RouterC(config-router)# neighbor 172.16.3.30 remote-as 200
RouterC(config-router)# neighbor 172.16.3.30 ebgp-multihop 1
RouterC(config-router)# address-family ipv4
RouterC(config-router-af)# bgp dmzlink-bw

RouterC(config-router-af)# neighbor 10.10.10.1 activate
RouterC(config-router-af)# neighbor 10.10.10.1 send-community both
RouterC(config-router-af)# neighbor 10.10.10.1 next-hop-self
RouterC(config-router-af)# neighbor 10.10.10.2 activate
RouterC(config-router-af)# neighbor 10.10.10.2 send-community both
RouterC(config-router-af)# neighbor 10.10.10.2 next-hop-self
RouterC(config-router-af)# neighbor 172.16.3.3 activate
RouterC(config-router-af)# neighbor 172.16.3.3 dmzlink-bw

RouterC(config-router-af)# maximum-paths ibgp 6
RouterC(config-router-af)# maximum-paths 6
```

Example: Verifying BGP Link Bandwidth

The examples in this section show the verification of this feature on Router A, Router B, and Router C.

Router B

In the following example, the **show ip bgp** command is entered on Router B to verify that two unequal cost best paths have been installed into the BGP routing table. The bandwidth for each link is displayed with each route.

```
RouterB# show ip bgp 192.168.1.0

BGP routing table entry for 192.168.1.0/24, version 48
Paths: (2 available, best #2)
Multipath: eBGP
  Advertised to update-groups:
     1          2
  200
    172.16.1.1 from 172.16.1.2 (192.168.1.1)
      Origin incomplete, metric 0, localpref 100, valid, external, multipath, best
      Extended Community: 0x0:0:0
```

```
      DMZ-Link Bw 278 kbytes
  200
    172.16.2.2 from 172.16.2.2 (192.168.1.1)
      Origin incomplete, metric 0, localpref 100, valid, external, multipath, best
      Extended Community: 0x0:0:0
      DMZ-Link Bw 625 kbytes
```

Router A

In the following example, the **show ip bgp** command is entered on Router A to verify that the link bandwidth extended community has been propagated through the iBGP network to Router A. Exit links are located on Router B and Router C. The output shows that a route for each exit link to autonomous system 200 has been installed as a best path in the BGP routing table.

```
RouterA# show ip bgp 192.168.1.0

BGP routing table entry for 192.168.1.0/24, version 48
Paths: (3 available, best #3)
Multipath: eBGP
  Advertised to update-groups:
     1          2
  200
    172.16.1.1 from 172.16.1.2 (192.168.1.1)
      Origin incomplete, metric 0, localpref 100, valid, external, multipath
      Extended Community: 0x0:0:0
      DMZ-Link Bw 278 kbytes
  200
    172.16.2.2 from 172.16.2.2 (192.168.1.1)
      Origin incomplete, metric 0, localpref 100, valid, external, multipath, best
      Extended Community: 0x0:0:0
      DMZ-Link Bw 625 kbytes
  200
    172.16.3.3 from 172.16.3.3 (192.168.1.1)
      Origin incomplete, metric 0, localpref 100, valid, external, multipath, best
      Extended Community: 0x0:0:0
      DMZ-Link Bw 2500 kbytes
```

Router A

In the following example, the **show ip route** command is entered on Router A to verify the multipath routes that are advertised and the associated traffic share values:

```
RouterA# show ip route 192.168.1.0

Routing entry for 192.168.1.0/24
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Last update from 172.168.1.1 00:01:43 ago
  Routing Descriptor Blocks:
  * 172.168.1.1, from 172.168.1.1, 00:01:43 ago
      Route metric is 0, traffic share count is 13
      AS Hops 1, BGP network version 0
      Route tag 200
    172.168.2.2, from 172.168.2.2, 00:01:43 ago
      Route metric is 0, traffic share count is 30
      AS Hops 1, BGP network version 0
      Route tag 200
    172.168.3.3, from 172.168.3.3, 00:01:43 ago
      Route metric is 0, traffic share count is 120
      AS Hops 1, BGP network version 0
      Route tag 200
```

Where to Go Next

For information about the BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN feature, see the "BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN" module in the *IP Routing: BGP Configuration Guide*.

For information about the iBGP Multipath Load Sharing feature, see the "iBGP Multipath Load Sharing" module in the *IP Routing: BGP Configuration Guide* .

Additional References

Related Documents

Related Topic	Document Title
BGP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
CEF configuration tasks	*IP Switching Cisco Express Forwarding Configuration Guide*

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFC	Title
draft-ramachandra-bgp-ext-communities-09.txt	*BGP Extended 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 Link Bandwidth

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 34: Feature Information for BGP Link Bandwidth

Feature Name	Releases	Feature Information
BGP Link Bandwidth	12.2(2)T 12.2(14)S	This feature advertises the bandwidth of an autonomous system exit link as an extended community. The link bandwidth extended community attribute is propagated to iBGP peers when extended community exchange is enabled. The following commands were introduced or modified: **router bgp, address-family ipv4, address-family ipv4**, **bgp dmzlink-bw, neighbor**, **show ip bgp**, **show ip route**.

CHAPTER 22

BGP NSF Awareness

Nonstop Forwarding (NSF) awareness allows a router 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 router that is running BGP to forward packets along routes that are already known for a router that is performing an SSO operation. This capability allows the BGP peers of the failing router to retain the routing information that is advertised by the failing router and continue to use this information until the failed router 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 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 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.

	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*] [**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

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.

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.
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*.

	Command or Action	Purpose
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 the peer sessions 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.

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:

```
Router> 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:

```
Router# 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. In Cisco IOS Releases 12.2(33)SRC, 12.2(33)SB, or later releases, the ability to enable or disable the BGP graceful restart capability for an individual BGP neighbor, peer group or peer session template was introduced and output was added to this command to show the BGP graceful restart status.

The following partial output example using a Cisco IOS Release 12.2(33)SRC image, displays the graceful restart information for internal BGP neighbor 172.21.1.2 at Router C in the figure above. Note the “Graceful-Restart is enabled” message.

Example:

```
Router# 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
```

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
```

Additional References

Related Documents

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

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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 http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 35: Feature Information for BGP NSF Awareness

Feature Name	Releases	Feature Information
BGP NSF Awareness	12.2(15)T	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. The following commands were introduced or modified: **bgp graceful-restart**, **show ip bgp**, **show ip bgp neighbors**.

CHAPTER 23

BGP Graceful Restart per Neighbor

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

Finding Feature Information

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

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

Information About BGP Graceful Restart per Neighbor

BGP Graceful Restart per Neighbor

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

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

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

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

BGP Peer Session Templates

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

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

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

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

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

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

How to Configure BGP Graceful Restart per Neighbor

Enabling BGP Graceful Restart for an Individual BGP Neighbor

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 unicast address family. By default, the router is placed in address family configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:**	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

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

Examples

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

```
Device# show ip bgp neighbors 172.21.1.2

BGP neighbor is 172.21.1.2,  remote AS 45000, internal link
  BGP version 4, remote router ID 172.22.1.1
  BGP state = Established, up for 00:01:01
  Last read 00:00:02, last write 00:00:07, hold time is 180, keepalive intervals
```

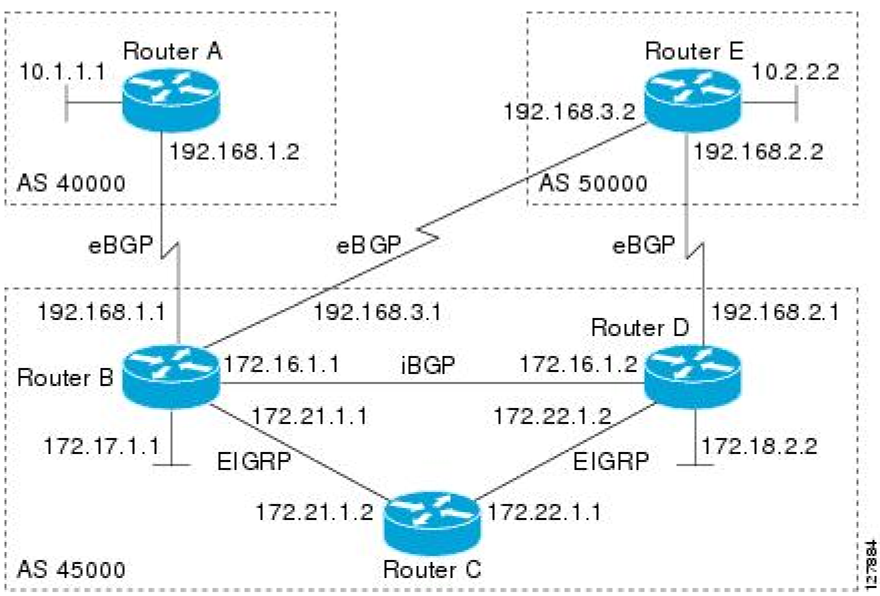
```
  Neighbor sessions:
    1 active, is multisession capable
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Address family IPv4 Unicast: advertised and received
    Graceful Restart Capability: advertised
    Multisession Capability: advertised and received
!
  Address tracking is enabled, the RIB does have a route to 172.21.1.2
  Connections established 1; dropped 0
  Last reset never
  Transport(tcp) path-mtu-discovery is enabled
  Graceful-Restart is enabled, restart-time 120 seconds, stalepath-time 360 secs
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
```

Enabling and Disabling BGP Graceful Restart Using BGP Peer Session Templates

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

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

Figure 41: 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.
Step 4	**template peer-session** *session-template-name* **Example:**	Enters session-template configuration mode and creates a peer session template.

	Command or Action	Purpose
	`Device(config-router)# template peer-session S1`	• 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.
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

	Command or Action	Purpose
		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*] **Example:** `Device# show ip bgp template peer-session`	(Optional) Displays locally configured peer session templates. • The output can be filtered to display a single peer policy template by using the *session-template-name* argument. This command also supports all standard output modifiers.
Step 17	**show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] |	(Optional) Displays information about TCP and BGP connections to neighbors.

	Command or Action	Purpose
	dampened-routes | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]] **Example:** `Device# show ip bgp neighbors 192.168.1.2`	• "Graceful Restart Capability: advertised" will be displayed for each neighbor that has exchanged graceful restart capabilities with this router. • In this example, the output is filtered to display information about the BGP peer at 192.168.1.2.

Examples

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

```
Device# show ip bgp neighbors 192.168.1.2

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

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

```
Device# show ip bgp neighbors 192.168.3.2

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

```
  Graceful-Restart is disabled
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
```

Disabling BGP Graceful Restart for a BGP Peer Group

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
		unicast address family if the **unicast** keyword is not specified. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **vrf** keyword and *vrf-name* argument specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router-af)# neighbor PG1 peer-group`	Creates a BGP peer group. • In this example, the peer group named PG1 is created.
Step 6	**neighbor** *peer-group-name* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor PG1 remote-as 45000`	Configures peering with a BGP peer group in the specified autonomous system. • In this example, the BGP peer group named PG1 is added to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 7	**neighbor** *peer-group-name* **ha-mode graceful-restart** [**disable**] **Example:** `Device(config-router-af)# neighbor PG1 ha-mode graceful-restart disable`	Enables the BGP graceful restart capability for a BGP neighbor. • Use the **disable** keyword to disable BGP graceful restart capability. • If you enter this command after the BGP session has been established, you must restart the session for the capability to be exchanged with the BGP neighbor. • In this example, the BGP graceful restart capability is disabled for the BGP peer group named PG1.
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**]]]	(Optional) Displays information about TCP and BGP connections to neighbors.

	Command or Action	Purpose
	Example: `Device# show ip bgp neighbors 172.16.1.2`	• In this example, the output is filtered to display information about the BGP peer at 172.16.1.2 and the "Graceful-Restart is disabled" line shows that the graceful restart capability is disabled for this neighbor.

Examples

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

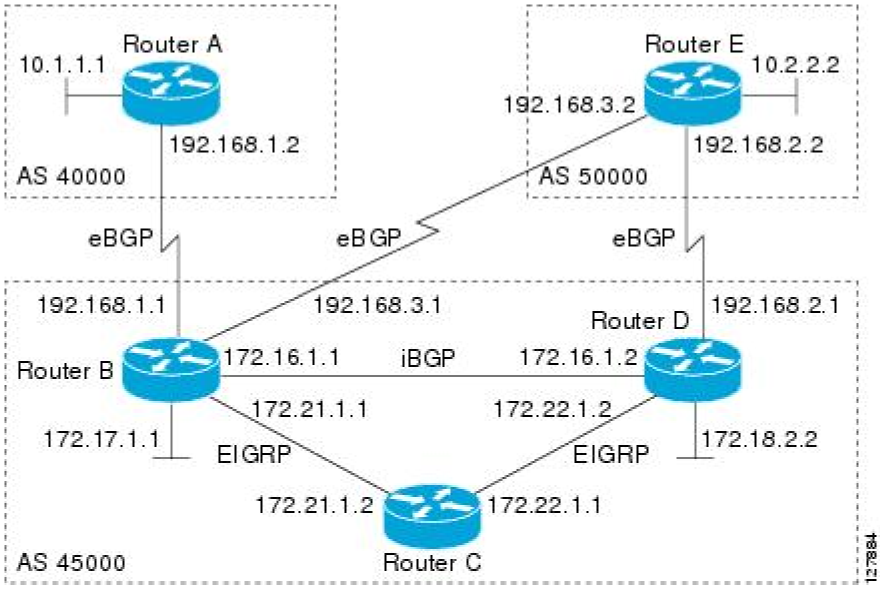
```
Device# show ip bgp neighbors 172.16.1.2

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

Configuration Examples for BGP Graceful Restart per Neighbor

Examples: Enabling and Disabling BGP Graceful Restart per Neighbor

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

Figure 42: Network Topology Showing BGP Neighbors for BGP Graceful Restart



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

```
router bgp 45000
 template peer-session S1
 remote-as 40000
 ha-mode graceful-restart
 exit-peer-session
 template peer-session S2
 remote-as 50000
 ha-mode graceful-restart disable
 exit-peer-session
 bgp log-neighbor-changes
 bgp graceful-restart restart-time 150
 bgp graceful-restart stalepath-time 400
 address-family ipv4 unicast
 neighbor PG1 peer-group
 neighbor PG1 remote-as 45000
 neighbor PG1 ha-mode graceful-restart disable
 neighbor 172.16.1.2 peer-group PG1
 neighbor 172.21.1.2 remote-as 45000
 neighbor 172.21.1.2 activate
 neighbor 172.21.1.2 ha-mode graceful-restart
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.1.2 inherit peer-session S1
 neighbor 192.168.3.2 remote-as 50000
 neighbor 192.168.3.2 inherit peer-session S2
 end
clear ip bgp *
```

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

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

Additional References

Related Documents

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

Standards and RFCs

Standard/RFC	Title
RFC 4724	*Graceful Restart Mechanism for BGP*

Technical Assistance

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

Feature Information for BGP Graceful Restart per Neighbor

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

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

Table 36: Feature Information for BGP Graceful Restart per Neighbor

Feature Name	Releases	Feature Information
BGP Graceful Restart per Neighbor	15.0(1)M	The BGP Graceful Restart per Neighbor feature enables or disables the BGP graceful restart capability for an individual BGP neighbor, including using peer session templates and BGP peer groups. The following commands were introduced by this feature: **ha-mode graceful-restart**, and**neighbor ha-mode graceful-restart**. The following command was modified by this feature: **show ip bgp neighbors**.

CHAPTER 24

BGP Support for BFD

Bidirectional Forwarding Detection (BFD) is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. Because the network administrator can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different routing protocol hello mechanisms, network profiling and planning will be easier, and reconvergence time will be consistent and predictable. The main benefit of implementing BFD for BGP is a significantly faster reconvergence time.

Finding Feature Information

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

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

Information About BGP Support for BFD

BFD for BGP

Bidirectional Forwarding Detection (BFD) is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. Because the network administrator can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different routing protocol hello mechanisms, network profiling and planning will be easier, and reconvergence time will be consistent and predictable. The main benefit of implementing BFD for BGP is a marked decrease in reconvergence time.

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

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

How to Decrease BGP Convergence Time Using BFD

Prerequisites

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

Restrictions

- For the Cisco implementation of BFD Support for BGP in Cisco IOS Release15.1(1)SG, only asynchronous mode is supported. In asynchronous mode, either BFD peer can initiate a BFD session.
- IPv6 encapsulation is supported.
- BFD multihop is supported.

Decreasing BGP Convergence Time Using BFD

You start a BFD process by configuring BFD on the interface. When the BFD process is started, no entries are created in the adjacency database, in other words, no BFD control packets are sent or received. The adjacency creation takes places once you have configured BFD support for the applicable routing protocols. The first two tasks must be configured to implement BFD support for BGP to reduce the BGP convergence time. The third task is an optional task to help monitor or troubleshoot BFD.

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

Configuring BFD Session Parameters on the Interface

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface** *type number*
4. **bfd interval** *milliseconds* **min_rx** *milliseconds* **multiplier** *interval-multiplier*
5. **end**

DETAILED STEPS

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

Configuring BFD Support for BGP

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

Before you begin

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp tag1`	Specifies a BGP process and enters router configuration mode.
Step 4	**neighbor** *ip-address* **fall-over bfd** **Example:** `Router(config-router)# neighbor 172.16.10.2 fall-over bfd`	Enables BFD support for fallover.
Step 5	**end** **Example:** `Router(config-router)# end`	Returns the router to privileged EXEC mode.
Step 6	**show bfd neighbors** [**details**] **Example:** `Router# show bfd neighbors detail`	Verifies that the BFD neighbor is active and displays the routing protocols that BFD has registered.
Step 7	**show ip bgp neighbors** [*ip-address* [**received-routes** | **routes** | **advertised-routes** | **paths** [*regexp*] | **dampened-routes** | **flap-statistics** | **received prefix-filter** | **policy** [**detail**]]]	Displays information about BGP and TCP connections to neighbors.

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

Monitoring and Troubleshooting BFD

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

SUMMARY STEPS

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

DETAILED STEPS

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

Additional References

Related Documents

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

Related Topic	Document Title
Configuring BFD support for another routing protocol	IP Routing: BFD Configuration Guide

Technical Assistance

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

Feature Information for BGP Support for BFD

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

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

Table 37: Feature Information for BGP Support for BFD

Feature Name	Releases	Feature Information
BGP Support for BFD	12.4(4)T	Bidirectional Forwarding Detection (BFD) is a detection protocol designed to provide fast forwarding path failure detection times for all media types, encapsulations, topologies, and routing protocols. In addition to fast forwarding path failure detection, BFD provides a consistent failure detection method for network administrators. Because the network administrator can use BFD to detect forwarding path failures at a uniform rate, rather than the variable rates for different routing protocol hello mechanisms, network profiling and planning will be easier, and reconvergence time will be consistent and predictable. The main benefit of implementing BFD for BGP is a significantly faster reconvergence time. The following commands were introduced or modified by this feature: **bfd**, **neighbor fall-over**, **show bfd neighbors**, and **show ip bgp neighbors**.

CHAPTER 25

iBGP Multipath Load Sharing

This feature module describes the iBGP Multipath Load Sharing feature.

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.

Restrictions for iBGP Multipath Load Sharing

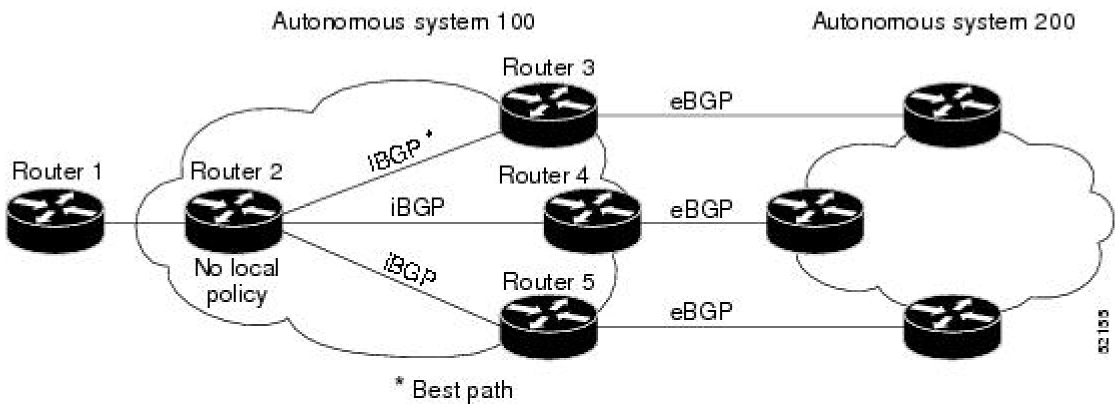
- Route Reflector Limitation—With multiple iBGP paths installed in a routing table, a route reflector will advertise only one of the paths (one next hop).
- Memory Consumption Restriction—Each IP routing table entry for a BGP prefix that has multiple iBGP paths uses approximately 350 bytes of additional memory. We recommend not using this feature on a router with a low amount of available memory and especially when the router is carrying a full Internet routing table.

Information about iBGP Multipath Load Sharing

iBGP Multipath Load Sharing Overview

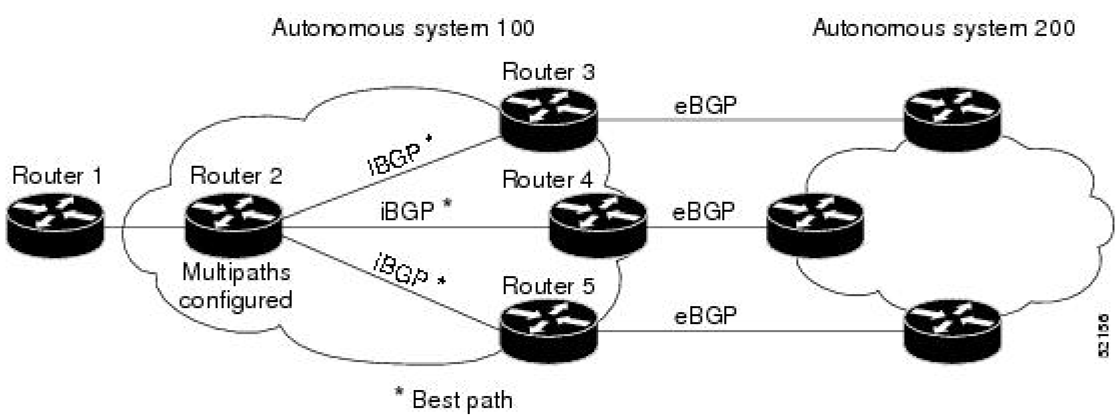
When a Border Gateway Protocol (BGP) speaking router with no local policy configured receives multiple network layer reachability information (NLRI) from the internal BGP (iBGP) for the same destination, the router will choose one iBGP path as the best path. The best path is then installed in the IP routing table of the router. For example, in the figure below, although there are three paths to autonomous system 200, Router 2 determines that one of the paths to autonomous system 200 is the best path and uses this path only to reach autonomous system 200.

Figure 43: Non-MPLS Topology with One Best Path



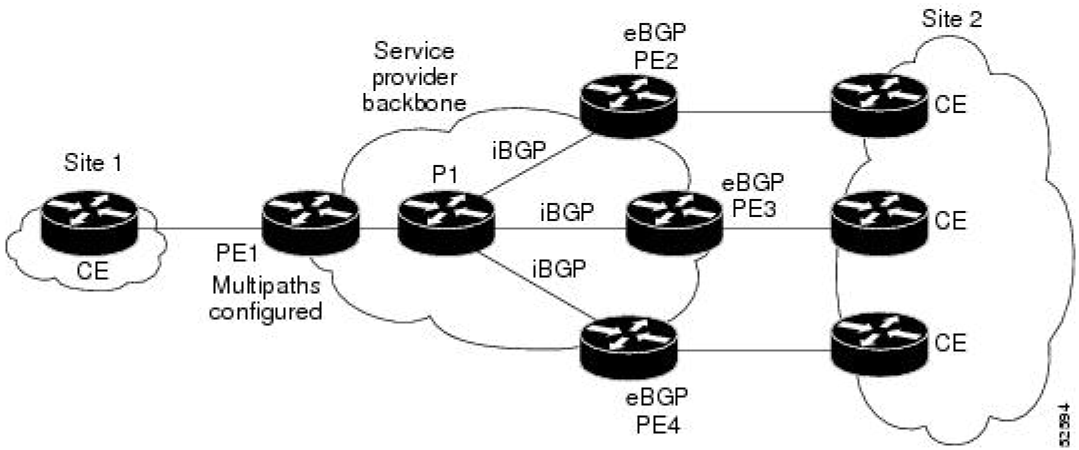
The iBGP Multipath Load Sharing feature enables the BGP speaking router to select multiple iBGP paths as the best paths to a destination. The best paths or multipaths are then installed in the IP routing table of the router. For example, on router 2 in the figure below, the paths to routers 3, 4, and 5 are configured as multipaths and can be used to reach autonomous system 200, thereby equally sharing the load to autonomous system 200.

Figure 44: Non-MPLS Topology with Three Multipaths



The iBGP Multipath Load Sharing feature functions similarly in a Multiprotocol Label Switching (MPLS) Virtual Private Network (VPN) with a service provider backbone. For example, on router PE1 in the figure below, the paths to routers PE2, PE3, and PE4 can be selected as multipaths and can be used to equally share the load to site 2.

Figure 45: MPLS VPN with Three Multipaths



For multiple paths to the same destination to be considered as multipaths, the following criteria must be met:

- All attributes must be the same. The attributes include weight, local preference, autonomous system path (entire attribute and not just length), origin code, Multi Exit Discriminator (MED), and Interior Gateway Protocol (IGP) distance.
- The next hop router for each multipath must be different.

Even if the criteria are met and multiple paths are considered multipaths, the BGP speaking router will still designate one of the multipaths as the best path and advertise this best path to its neighbors.

The iBGP Multipath Load Sharing feature is similar to BGP multipath support for external BGP (eBGP) paths; however, the iBGP Multipath Load Sharing feature is applied to internal rather than eBGP paths.

Benefits of iBGP Multipath Load Sharing

Configuring multiple iBGP best paths enables a router to evenly share the traffic destined for a particular site.

How to Configure IBGP Multipath Load Sharing

Configuring iBGP Multipath Load Sharing

To configure the iBGP Multipath Load Sharing feature, use the following command in router configuration mode:

Command	Purpose
`Device(config-router)#` **`maximum-paths ibgp`** *`maximum-number`*	Controls the maximum number of parallel iBGP routes that can be installed in a routing table.

Verifying iBGP Multipath Load Sharing

To verify that the iBGP Multipath Load Sharing feature is configured correctly, perform the following steps:

SUMMARY STEPS

1. Enter the **show ip bgp** *network-number* EXEC command to display attributes for a network in a non-MPLS topology, or the **show ip bgp vpnv4 all** *ip-prefix* EXEC command to display attributes for a network in an MPLS VPN:
2. In the display resulting from the **show ip bgp** *network-number* EXEC command or the **show ip bgp vpnv4 all** *ip-prefix* EXEC command, verify that the intended multipaths are marked as "multipaths." Notice that one of the multipaths is marked as "best."
3. Enter the **show ip route** *ip-address* EXEC command to display routing information for a network in a non-MPLS topology or the **show ip route vrf** *vrf-name ip-prefix* EXEC command to display routing information for a network in an MPLS VPN:
4. Verify that the paths marked as "multipath" in the display resulting from the **show ip bgp** *ip-prefix* EXEC command or the **show ip bgp vpnv4 all** *ip-prefix* EXEC command are included in the routing information. (The routing information is displayed after performing Step 3.)

DETAILED STEPS

Step 1 Enter the **show ip bgp** *network-number* EXEC command to display attributes for a network in a non-MPLS topology, or the **show ip bgp vpnv4 all** *ip-prefix* EXEC command to display attributes for a network in an MPLS VPN:

Example:

```
Device# show ip bgp 10.22.22.0

BGP routing table entry for 10.22.22.0/24, version 119
Paths:(6 available, best #1)
Multipath:iBGP
Flag:0x820
  Advertised to non peer-group peers:
  10.1.12.12
  22
    10.2.3.8 (metric 11) from 10.1.3.4 (100.0.0.5)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath, best
      Originator:100.0.0.5, Cluster list:100.0.0.4
  22
    10.2.1.9 (metric 11) from 10.1.1.2 (100.0.0.9)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Originator:100.0.0.9, Cluster list:100.0.0.2
  22
    10.2.5.10 (metric 11) from 10.1.5.6 (100.0.0.10)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Originator:100.0.0.10, Cluster list:100.0.0.6
  22
    10.2.4.10 (metric 11) from 10.1.4.5 (100.0.0.10)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Originator:100.0.0.10, Cluster list:100.0.0.5
  22
    10.2.6.10 (metric 11) from 10.1.6.7 (100.0.0.10)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Originator:100.0.0.10, Cluster list:100.0.0.7

Device# show ip bgp vpnv4 all 10.22.22.0
```

```
BGP routing table entry for 100:1:10.22.22.0/24, version 50
Paths:(6 available, best #1)
Multipath:iBGP
  Advertised to non peer-group peers:
  200.1.12.12
  22
    10.22.7.8 (metric 11) from 10.11.3.4 (100.0.0.8)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath, best
      Extended Community:RT:100:1
      Originator:100.0.0.8, Cluster list:100.1.1.44
  22
    10.22.1.9 (metric 11) from 10.11.1.2 (100.0.0.9)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:RT:100:1
      Originator:100.0.0.9, Cluster list:100.1.1.22
  22
    10.22.6.10 (metric 11) from 10.11.6.7 (100.0.0.10)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:RT:100:1
      Originator:100.0.0.10, Cluster list:100.0.0.7
  22
    10.22.4.10 (metric 11) from 10.11.4.5 (100.0.0.10)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:RT:100:1
      Originator:100.0.0.10, Cluster list:100.0.0.5
  22
    10.22.5.10 (metric 11) from 10.11.5.6 (100.0.0.10)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:RT:100:1
      Originator:100.0.0.10, Cluster list:100.0.0.6
```

Step 2 In the display resulting from the **show ip bgp** *network-number* EXEC command or the **show ip bgp vpnv4 all** *ip-prefix* EXEC command, verify that the intended multipaths are marked as “multipaths." Notice that one of the multipaths is marked as “best.”

Step 3 Enter the **show ip route** *ip-address* EXEC command to display routing information for a network in a non-MPLS topology or the **show ip route vrf** *vrf-name ip-prefix* EXEC command to display routing information for a network in an MPLS VPN:

Example:

```
Device# show ip route 10.22.22.0

Routing entry for 10.22.22.0/24
  Known via "bgp 1", distance 200, metric 0
  Tag 22, type internal
  Last update from 10.2.6.10 00:00:03 ago
  Routing Descriptor Blocks:
  * 10.2.3.8, from 10.1.3.4, 00:00:03 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.2.1.9, from 10.1.1.2, 00:00:03 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.2.5.10, from 10.1.5.6, 00:00:03 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.2.4.10, from 10.1.4.5, 00:00:03 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.2.6.10, from 10.1.6.7, 00:00:03 ago
      Route metric is 0, traffic share count is 1
```

```
      AS Hops 1

Device# show ip route vrf PATH 10.22.22.0

Routing entry for 10.22.22.0/24
  Known via "bgp 1", distance 200, metric 0
  Tag 22, type internal
  Last update from 10.22.5.10 00:01:07 ago
  Routing Descriptor Blocks:
  * 10.22.7.8 (Default-IP-Routing-Table), from 10.11.3.4, 00:01:07 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.22.1.9 (Default-IP-Routing-Table), from 10.11.1.2, 00:01:07 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.22.6.10 (Default-IP-Routing-Table), from 10.11.6.7, 00:01:07 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.22.4.10 (Default-IP-Routing-Table), from 10.11.4.5, 00:01:07 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.22.5.10 (Default-IP-Routing-Table), from 10.11.5.6, 00:01:07 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
```

Step 4 Verify that the paths marked as "multipath” in the display resulting from the **show ip bgp** *ip-prefix* EXEC command or the **show ip bgp vpnv4 all** *ip-prefix* EXEC command are included in the routing information. (The routing information is displayed after performing Step 3.)

Monitoring and Maintaining iBGP Multipath Load Sharing

To display iBGP Multipath Load Sharing information, use the following commands in EXEC mode, as needed:

Command	Purpose
Device# **show ip bgp** *ip-prefix*	Displays attributes and multipaths for a network in a non-MPLS topology.
Device# **show ip bgp vpnv4 all** *ip-prefix*	Displays attributes and multipaths for a network in an MPLS VPN.
Device# **show ip route** *ip-prefix*	Displays routing information for a network in a non-MPLS topology.
Device# **show ip route vrf** *vrf-name ip-prefix*	Displays routing information for a network in an MPLS VPN.

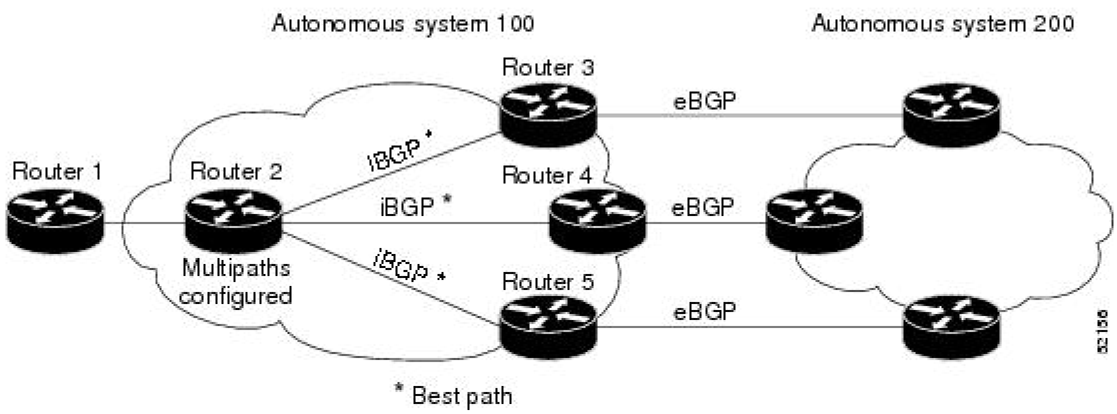
Configuration Examples for iBGP Multipath Load Sharing

Example: iBGP Multipath Load Sharing in a Non-MPLS Topology

Both examples assume that the appropriate attributes for each path are equal and that the next hop router for each multipath is different.

The following example shows how to set up the iBGP Multipath Load Sharing feature in a non-MPLS topology (see the figure below).

Figure 46: Non-MPLS Topology Example



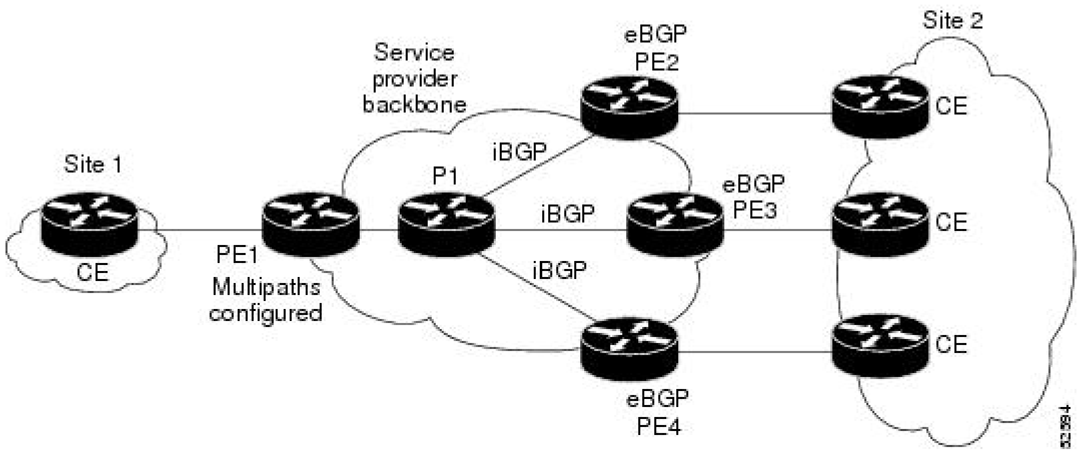
Router 2 Configuration

```
router bgp 100
maximum-paths ibgp 3
```

Example: iBGP Multipath Load Sharing in an MPLS VPN Topology

The following example shows how to set up the iBGP Multipath Load Sharing feature in an MPLS VPN topology (see the figure below).

Figure 47: MPLS VPN Topology Example



Router PE1 Configuration

```
router bgp 100
address-family ipv4 unicast vrf site2
 maximum-paths ibgp 3
```

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
BGP multipath load sharing for both eBGP and iBGP in an MPLS-VPN	"BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN" module in the *IP Routing: BGP Configuration Guide*
Advertising the bandwidth of an autonomous system exit link as an extended community	"BGP Link Bandwidth" module in the *IP Routing: BGP Configuration Guide*

Standards

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

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Technical Assistance

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

Feature Information for iBGP Multipath Load Sharing

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 38: Feature Information for iBGP Multipath Load Sharing

Feature Name	Releases	Feature Information
iBGP Multipath Load Sharing	12.2(14)S 12.2(2)T	The iBGP Multipath Load Sharing feature enables the BGP speaking router to select multiple iBGP paths as the best paths to a destination. The following command was introduced: **maximum-paths ibgp** The following commands were modified: **show ip bgp**, **show ip bgp vpnv4**, **show ip route**, and **show ip route vrf**.

CHAPTER 26

BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

The BGP Multipath Load Sharing for eBGP and iBGP feature allows you to configure multipath load balancing with both external BGP (eBGP) and internal BGP (iBGP) paths in Border Gateway Protocol (BGP) networks that are configured to use Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPNs). This feature provides improved load balancing deployment and service offering capabilities and is useful for multi-homed autonomous systems and Provider Edge (PE) routers that import both eBGP and iBGP paths from multihomed and stub networks.

Finding Feature Information

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

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

Prerequisites for BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Load Balancing is Configured Under CEF

Cisco Express Forwarding (CEF) or distributed CEF (dCEF) must be enabled on all participating routers.

Restrictions for BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Address Family Support

This feature is configured on a per VPN routing and forwarding instance (VRF) basis. This feature can be configured under only the IPv4 VRF address family.

Memory Consumption Restriction

Each BGP multipath routing table entry will use additional memory. We recommend that you do not use this feature on a router with a low amount of available memory and especially if router is carries full Internet routing tables.

Route Reflector Limitation

When multiple iBGP paths installed in a routing table, a route reflector will advertise only one paths (next hop). If a router is behind a route reflector, all routers that are connected to multihomed sites will not be advertised unless a different route distinguisher is configured for each VRF.

Information About BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Multipath Load Sharing Between eBGP and iBGP

A BGP routing process will install a single path as the best path in the routing information base (RIB) by default. The **maximum-paths** command allows you to configure BGP to install multiple paths in the RIB for multipath load sharing. BGP uses the best path algorithm to still select a single multipath as the best path and advertise the best path to BGP peers.

Note The number of paths of multipaths that can be configured is documented on the **maximum-paths** command reference page.

Load balancing over the multipaths is performed by CEF. CEF load balancing is configured on a per-packet round robin or on a per session (source and destination pair) basis. For information about CEF, refer to the "Cisco Express Forwarding Overview" documentation:

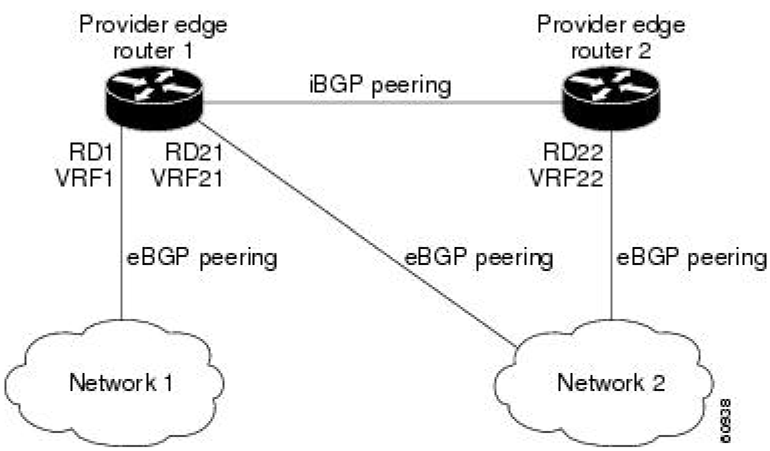
The BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature is enabled only under the IPv4 VRF address family configuration mode. When enabled, this feature can perform load balancing on eBGP and/or iBGP paths that are imported into the VRF. The number of multipaths is configured on a per VRF basis. Separate VRF multipath configurations are isolated by unique route distinguisher.

Note The BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature operates within the parameters of configured outbound routing policy.

eBGP and iBGP Multipath Load Sharing in a BGP MPLS Network

The figure below shows a service provider BGP MPLS network that connects two remote networks to PE router 1 and PE router 2. PE router 1 and PE router 2 are both configured for VPNv4 unicast iBGP peering. Network 2 is a multihomed network that is connected to PE router 1 and PE router 2. Network 2 also has extranet VPN services configured with Network 1. Both Network 1 and Network 2 are configured for eBGP peering with the PE routers.

Figure 48: A Service Provider BGP MPLS Network

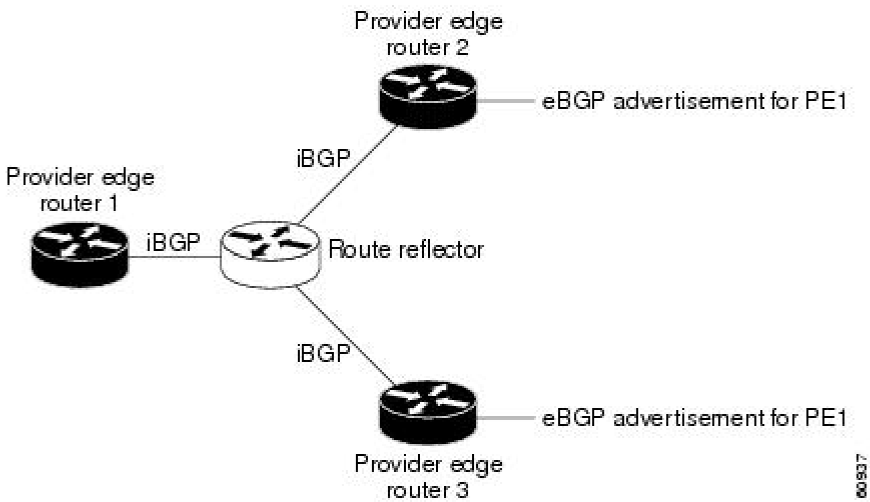


PE router 1 can be configured with the BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature so that both iBGP and eBGP paths can be selected as multipaths and imported into the VRF of Network 1. The multipaths will be used by CEF to perform load balancing. IP traffic that is sent from Network 2 to PE router 1 and PE router 2 will be sent across the eBGP paths as IP traffic. IP traffic that is sent across the iBGP path will be sent as MPLS traffic, and MPLS traffic that is sent across an eBGP path will be sent as IP traffic. Any prefix that is advertised from Network 2 will be received by PE router 1 through route distinguisher (RD) 21 and RD 22.The advertisement through RD 21 will be carried in IP packets, and the advertisement through RD 22 will be carried in MPLS packets. Both paths can be selected as multipaths for VRF1 and installed into the VRF1 RIB.

eBGP and iBGP Multipath Load Sharing With Route Reflectors

The figure below shows a topology that contains three PE routers and a route reflector, all configured for iBGP peering. PE router 2 and PE router 3 each advertise an equal preference eBGP path to PE router 1. By default, the route reflector will choose only one path and advertise PE router 1.

Figure 49: A Topology with a Route Reflector



For all equal preference paths to PE router 1 to be advertised through the route reflector, you must configure each VRF with a different RD. The prefixes received by the route reflector will be recognized differently and advertised to PE router 1.

Benefits of Multipath Load Sharing for Both eBGP and iBGP

The BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS VPN feature allows multihomed autonomous systems and PE routers to be configured to distribute traffic across both eBGP and iBGP paths.

How to Configure BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Configuring Multipath Load Sharing for Both eBGP and iBGP

To configure this feature, perform the steps in this section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4 vrf** *vrf-name*
5. **maximum-paths eibgp** *number*

6. **end**

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 40000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**address-family ipv4 vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf RED`	Places the router in address family configuration mode. • Separate VRF multipath configurations are isolated by unique route distinguisher.
Step 5	**maximum-paths eibgp** *number* **Example:** `Device(config-router-af)# maximum-paths eibgp 6`	Configures the number of parallel iBGP and eBGP routes that can be installed into a routing table. **Note** The **maximum-paths eibgp** command can be configured only under the IPv4 VRF address family configuration mode and cannot be configured in any other address family configuration mode.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode, and enters privileged EXEC mode.

Verifying Multipath Load Sharing for Both eBGP an iBGP

SUMMARY STEPS

1. **enable**
2. **show ip bgp neighbors** [*neighbor-address* [**advertised-routes** | **dampened-routes** | **flap-statistics** | **paths** [*regexp*] | **received** *prefix-filter* | **received-routes** | **routes**]]
3. **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*}

4. **show ip route vrf** *vrf-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	**show ip bgp neighbors** [*neighbor-address* [**advertised-routes** | **dampened-routes** | **flap-statistics** | **paths** [*regexp*] | **received** *prefix-filter* | **received-routes** | **routes**]] **Example:** `Device# show ip bgp neighbors`	Displays information about the TCP and BGP connections to neighbors.
Step 3	**show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} **Example:** `Device# show ip bgp vpnv4 vrf RED`	Displays VPN address information from the BGP table. This command is used to verify that the VRF has been received by BGP.
Step 4	**show ip route vrf** *vrf-name* **Example:** `Device# show ip route vrf RED`	Displays the IP routing table associated with a VRF instance. The show ip route vrf command is used to verify that the VRF is in the routing table.

Configuration Examples for the BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN Feature

Example: Configuring eBGP and iBGP Multipath Load Sharing

This following configuration example configures a router in address-family mode to select six BGP routes (eBGP or iBGP) as multipaths:

```
Device(config)# router bgp 40000
 Device(config-router)# address-family ipv4 vrf RED
 Device(config-router-af)# maximum-paths eibgp 6
 Device(config-router-af)# end
```

Example: Verifying eBGP and iBGP Multipath Load Sharing

To verify that iBGP and eBGP routes have been configured for load sharing, use the **show ip bgp vpnv4** EXEC command or the **show ip route vrf** EXEC command.

In the following example, the **show ip bgp vpnv4** command is entered to display multipaths installed in the VPNv4 RIB:

```
Device# show ip bgp vpnv4 all 10.22.22.0

BGP routing table entry for 10:1:22.22.22.0/24, version 19
Paths:(5 available, best #5)
Multipath:eiBGP
  Advertised to non peer-group peers:
  10.0.0.2 10.0.0.3 10.0.0.4 10.0.0.5
  22
    10.0.0.2 (metric 20) from 10.0.0.4 (10.0.0.4)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:0x0:0:0 RT:100:1 0x0:0:0
      Originator:10.0.0.2, Cluster list:10.0.0.4
  22
    10.0.0.2 (metric 20) from 10.0.0.5 (10.0.0.5)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:0x0:0:0 RT:100:1 0x0:0:0
      Originator:10.0.0.2, Cluster list:10.0.0.5
  22
    10.0.0.2 (metric 20) from 10.0.0.2 (10.0.0.2)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:RT:100:1 0x0:0:0
  22
    10.0.0.2 (metric 20) from 10.0.0.3 (10.0.0.3)
      Origin IGP, metric 0, localpref 100, valid, internal, multipath
      Extended Community:0x0:0:0 RT:100:1 0x0:0:0
      Originator:10.0.0.2, Cluster list:10.0.0.3
  22
    10.1.1.12 from 10.1.1.12 (10.22.22.12)
      Origin IGP, metric 0, localpref 100, valid, external, multipath, best
      Extended Community:RT:100:1
```

In the following example, the **show ip route vrf** command is entered to display multipath routes in the VRF table:

```
Device# show ip route vrf PATH 10.22.22.0

Routing entry for 10.22.22.0/24
  Known via "bgp 1", distance 20, metric 0
  Tag 22, type external
  Last update from 10.1.1.12 01:59:31 ago
  Routing Descriptor Blocks:
  * 10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.4, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.5, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.2, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
    10.0.0.2 (Default-IP-Routing-Table), from 10.0.0.3, 01:59:31 ago
      Route metric is 0, traffic share count is 1
      AS Hops 1
```

```
10.1.1.12, from 10.1.1.12, 01:59:31 ago
  Route metric is 0, traffic share count is 1
  AS Hops 1
```

Where to Go Next

For information about advertising the bandwidth of an autonomous system exit link as an extended community, refer to the “BGP Link Bandwidth” module.

Additional References

Related Documents

Related Topic	Document Title
BGP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
Comprehensive BGP link bandwidth configuration examples and tasks	“BGP Link Bandwidth” module in the *IP Routing: BGP Configuration Guide*
CEF configuration tasks	“CEF Overview” module in the *IP Switching Cisco Express Forwarding Configuration Guide*

Standards

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

MIBs

MIBs	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
RFC 1771	*A Border Gateway Protocol 4 (BGP4)*
RFC 2547	*BGP/MPLS VPNs*

RFCs	Title
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 Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

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 39: Feature Information for BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN

Feature Name	Releases	Feature Configuration Information
BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN	12.0(24)S 12.2(14)S 12.2(18)SXE 12.2(4)T 15.0(1)S Cisco IOS XE Release 3.1.0SG	The BGP Multipath Load Sharing for eBGP and iBGP feature allows you to configure multipath load balancing with both eBGP and iBGP paths in BGP networks that are configured to use MPLS VPNs. This feature provides improved load balancing deployment and service offering capabilities and is useful for multi-homed autonomous systems and PE routers that import both eBGP and iBGP paths from multihomed and stub networks. The following command was introduced or modified by this feature: **maximum-paths eibgp.**

CHAPTER 27

Loadsharing IP Packets Over More Than Six Parallel Paths

The Loadsharing IP Packets Over More Than Six Parallel Paths feature increases the maximum number of parallel routes that can be installed to the routing table for multipath loadsharing.

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.

Overview of Loadsharing IP Packets over More Than Six Parallel Paths

The Loadsharing IP Packets over More Than Six Parallel Paths feature increases the maximum number of parallel routes that can be installed to the routing table. The maximum number has been increased from six to sixteen for the following commands:

- **maximum-paths**
- **maximum-paths eibgp**
- **maximum-paths ibgp**

The output of the **show ip route summary** command has been updated to display the number of parallel routes supported by the routing table.

The benefits of this feature include the following:

- More flexible configuration of parallel routes in the routing table.
- Ability to configure multipath loadsharing over more links to allow for the configuration of higher-bandwidth aggregation using lower-speed links.

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
eBGP multipath load sharing	"BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN" module
iBGP multipath load sharing	"iBGP Multipath Load Sharing" module

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE 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 Loadsharing IP Packets Over More Than Six Parallel Paths

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

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

Table 40: Feature Information for Loadsharing IP Packets Over More Than Six Parallel Paths

Feature Name	Releases	Feature Information
Loadsharing IP Packets Over More Than Six Parallel Paths	12.3(2)T 12.2(25)S Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.1.0SG	The Loadsharing IP Packets Over More Than Six Parallel Paths feature increases the maximum number of parallel routes that can be installed to the routing table for multipath loadsharing. This feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. The following commands were modified: • **maximum-paths** • **maximum-paths eibgp** • **maximum-paths ibgp** • **show ip route summary**

CHAPTER 28

BGP Policy Accounting

Border Gateway Protocol (BGP) policy accounting measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting is enabled on an input interface, and counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.

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

Before using the BGP Policy Accounting feature, you must enable BGP and CEF or dCEF on the router.

Information About BGP Policy Accounting

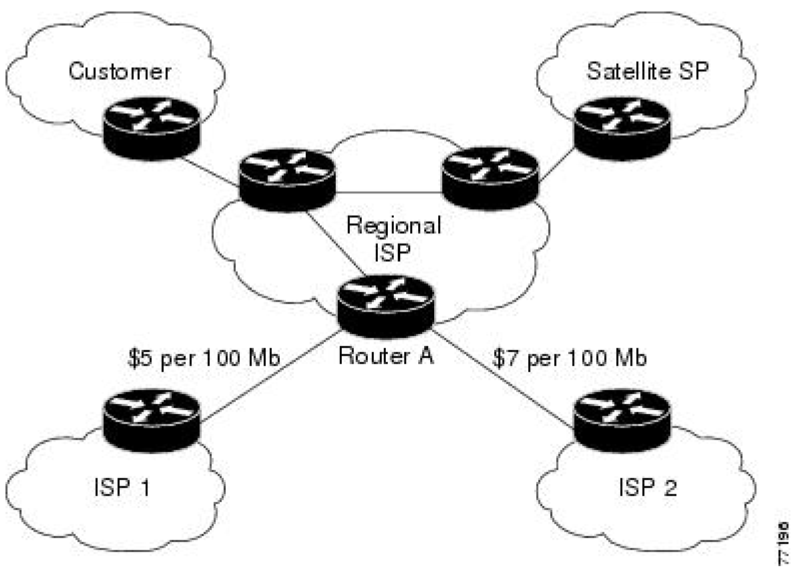
BGP Policy Accounting Overview

Border Gateway Protocol (BGP) policy accounting measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting is enabled on an input interface, and counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.

Using the BGP **table-map** command, prefixes added to the routing table are classified by BGP attribute, autonomous system number, or autonomous system path. Packet and byte counters are incremented per input interface. A Cisco IOS policy-based classifier maps the traffic into one of eight possible buckets, representing different traffic classes.

Using BGP policy accounting, you can account for traffic according to the route it traverses. Service providers (SPs) can identify and account for all traffic by customer and bill accordingly. In the figure below, BGP policy accounting can be implemented in Router A to measure packet and byte volumes in autonomous system buckets. Customers are billed appropriately for traffic that is routed from a domestic, international, or satellite source.

Figure 50: Sample Topology for BGP Policy Accounting



BGP policy accounting using autonomous system numbers can be used to improve the design of network circuit peering and transit agreements between Internet service providers (ISPs).

Benefits of BGP Policy Accounting

Account for IP Traffic Differentially

BGP policy accounting classifies IP traffic by autonomous system number, autonomous system path, or community list string, and increments packet and byte counters. Service providers can account for traffic and apply billing, according to the route specific traffic traverses.

Efficient Network Circuit Peering and Transit Agreement Design

Implementing BGP policy accounting on an edge router can highlight potential design improvements for peering and transit agreements.

How to Configure BGP Policy Accounting

Specifying the Match Criteria for BGP Policy Accounting

The first task in configuring BGP policy accounting is to specify the criteria that must be matched. Community lists, autonomous system paths, or autonomous system numbers are examples of BGP attributes that can be specified and subsequently matched using a route map.

To specify the BGP attribute to use for BGP policy accounting and create the match criteria in a route map, use the following commands in global configuration mode:

SUMMARY STEPS

1. Device(config)# **ip community-list** *community-list-number* {**permit** | **deny**} *community-number*
2. Device(config)# **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
3. Device(config-route-map)# **match community-list** *community-list-number* [**exact**]
4. Device(config-route-map)# **set traffic-index** *bucket-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **ip community-list** *community-list-number* {**permit** | **deny**} *community-number*	Creates a community list for BGP and controls access to it. This step must be repeated for each community to be specified.
Step 2	Device(config)# **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]	Enters route-map configuration mode and defines the conditions for policy routing. The *map-name* argument identifies a route map. The optional **permit** and **deny** keywords work with the match and set criteria to control how the packets are accounted for.

	Command or Action	Purpose
		The optional *sequence-number* argument indicates the position a new route map is to have in the list of route maps already configured with the same name.
Step 3	Device(config-route-map)# **match community-list** *community-list-number* [**exact**]	Matches a BGP community.
Step 4	Device(config-route-map)# **set traffic-index** *bucket-number*	Indicates where to output packets that pass a match clause of a route map for BGP policy accounting.

Classifying the IP Traffic and Enabling BGP Policy Accounting

After a route map has been defined to specify match criteria, you must configure a way to classify the IP traffic before enabling BGP policy accounting.

Using the **table-map** command, BGP classifies each prefix it adds to the routing table based on the match criteria. When the **bgp-policy accounting** command is configured on an interface, BGP policy accounting is enabled.

To classify the IP traffic and enable BGP policy accounting, use the following commands beginning in global configuration mode:

SUMMARY STEPS

1. Device(config)# **router bgp** *as-number*
2. Device(config-router)# **table-map** *route-map-name*
3. Device(config-router)# **network** *network-number* [**mask** *network-mask*]
4. Device(config-router)# **neighbor** *ip-address* **remote-as** *as-number*
5. Device(config-router)# **exit**
6. Device(config)# **interface** *interface-type interface-number*
7. Device(config-if)# **no ip directed-broadcast**
8. Device(config-if)# **ip address** *ip-address mask*
9. Device(config-if)# **bgp-policy accounting**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **router bgp** *as-number*	Configures a BGP routing process and enters router configuration mode for the specified routing process.
Step 2	Device(config-router)# **table-map** *route-map-name*	Classifies BGP prefixes entered in the routing table.
Step 3	Device(config-router)# **network** *network-number* [**mask** *network-mask*]	Specifies a network to be advertised by the BGP routing process.
Step 4	Device(config-router)# **neighbor** *ip-address* **remote-as** *as-number*	Specifies a BGP peer by adding an entry to the BGP routing table.
Step 5	Device(config-router)# **exit**	Exits to global configuration mode.

	Command or Action	Purpose
Step 6	Device(config)# **interface** *interface-type interface-number*	Specifies the interface type and number and enters interface configuration mode.
Step 7	Device(config-if)# **no ip directed-broadcast**	Configures the interface to drop directed broadcasts destined for the subnet to which that interface is attached, rather than being broadcast. This is a security issue.
Step 8	Device(config-if)# **ip address** *ip-address mask*	Configures the interface with an IP address.
Step 9	Device(config-if)# **bgp-policy accounting**	Enables BGP policy accounting for the interface.

Verifying BGP Policy Accounting

To verify that BGP policy accounting is operating, perform the following steps:

SUMMARY STEPS

1. Enter the **show ip cef** EXEC command with the **detail** keyword to learn which accounting bucket is assigned to a specified prefix.
2. Enter the **show ip bgp** EXEC command for the same prefix used in Step 1--192.168.5.0-- to learn which community is assigned to this prefix.
3. Enter the **show cef interface policy-statistics** EXEC command to display the per-interface traffic statistics.

DETAILED STEPS

Step 1 Enter the **show ip cef** EXEC command with the **detail** keyword to learn which accounting bucket is assigned to a specified prefix.

In this example, the output is displayed for the prefix 192.168.5.0. It shows that the accounting bucket number 4 (traffic_index 4) is assigned to this prefix.

Example:

```
Device# show ip cef 192.168.5.0 detail

192.168.5.0/24, version 21, cached adjacency to POS7/2
0 packets, 0 bytes, traffic_index 4
  via 10.14.1.1, 0 dependencies, recursive
   next hop 10.14.1.1, POS7/2 via 10.14.1.0/30
   valid cached adjacency
```

Step 2 Enter the **show ip bgp** EXEC command for the same prefix used in Step 1--192.168.5.0-- to learn which community is assigned to this prefix.

In this example, the output is displayed for the prefix 192.168.5.0. It shows that the community of 100:197 is assigned to this prefix.

Example:

```
Device# show ip bgp 192.168.5.0

BGP routing table entry for 192.168.5.0/24, version 2
```

```
Paths: (1 available, best #1)
  Not advertised to any peer
  100
    10.14.1.1 from 10.14.1.1 (32.32.32.32)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 100:197
```

Step 3 Enter the **show cef interface policy-statistics** EXEC command to display the per-interface traffic statistics.

In this example, the output shows the number of packets and bytes that have been assigned to each accounting bucket:

Example:

```
Device# show cef interface policy-statistics

POS7/0 is up (if_number 8)
Bucket    Packets              Bytes
1          0                     0
2          0                     0
3          50                    5000
4          100                   10000
5          100                   10000
6          10                    1000
7          0                     0
8          0                     0
```

Monitoring and Maintaining BGP Policy Accounting

Command	Purpose
`Device#` **`show cef interface`** `[`*`type number`*`]` **`policy-statistics`**	(Optional) Displays detailed CEF policy statistical information for all interfaces.
`Device#` **`show ip bgp`** `[`*`network`*`] [`*`network mask`*`] [`**`longer-prefixes`**`]`	(Optional) Displays entries in the BGP routing table.
`Device#` **`show ip cef`** `[`*`network`* `[`*`mask`*`]] [`**`detail`**`]`	(Optional) Displays entries in the Forwarding Information Base (FIB) or FIB summary information.

Configuration Examples for BGP Policy Accounting

Example: Specifying the Match Criteria for BGP Policy Accounting

In the following example, BGP communities are specified in community lists, and a route map named set_bucket is configured to match each of the community lists to a specific accounting bucket using the **set traffic-index** command:

```
ip community-list 30 permit 100:190
```

```
ip community-list 40 permit 100:198
ip community-list 50 permit 100:197
ip community-list 60 permit 100:296
!
route-map set_bucket permit 10
match community 30
set traffic-index 2
!
route-map set_bucket permit 20
match community 40
set traffic-index 3
!
route-map set_bucket permit 30
match community 50
set traffic-index 4
!
route-map set_bucket permit 40
match community 60
set traffic-index 5
```

Example: Classifying the IP Traffic and Enabling BGP Policy Accounting

In the following example, BGP policy accounting is enabled on POS interface 7/0 and the **table-map** command is used to modify the bucket number when the IP routing table is updated with routes learned from BGP:

```
router bgp 65000
 table-map set_bucket
 network 10.15.1.0 mask 255.255.255.0
 neighbor 10.14.1.1 remote-as 65100
!
ip classless
ip bgp-community new-format
!
interface POS7/0
 ip address 10.15.1.2 255.255.255.0
 no ip directed-broadcast
 bgp-policy accounting
 no keepalive
 crc 32
 clock source internal
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
Cisco Express Forwarding (CEF) and distributed CEF (dCEF) commands	Cisco IOS IP Switching Command Reference
Cisco Express Forwarding (CEF) and distributed CEF (dCEF) configuration information	"CEF Overview" module of the *Cisco IOS Switching Services Configuration Guide*

MIBs

MIB	MIBs Link
• CISCO-BGP-POLICY-ACCOUNTING-MIB **Note** CISCO-BGP-POLICY-ACCOUNTING-MIB is only available in the Cisco IOS Release 12.0(9)S, 12.0(17)ST, and later releases. This MIB is not available on any mainline and T-train release.	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 Policy Accounting

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 41: Feature Information for BGP Policy Accounting

Feature Name	Releases	Feature Information
BGP Policy Accounting	12.0(9)S 12.0(17)ST 12.2(13)T 15.0(1)S 12.2(50)SY Cisco IOS XE Release 3.8S	Border Gateway Protocol (BGP) policy accounting measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting is enabled on an input interface, and counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic. The following commands were introduced or modified: • **bgp-policy** • **set traffic-index** • **show cef interface policy-statistics** • **show ip bgp** • **show ip cef**

CHAPTER 29

BGP Policy Accounting Output Interface Accounting

Border Gateway Protocol (BGP) policy accounting (PA) measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting was previously available on an input interface only. The BGP Policy Accounting Output Interface Accounting feature introduces several extensions to enable BGP PA on an output interface and to include accounting based on a source address for both input and output traffic on an interface. Counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.

Finding Feature Information

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

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

Prerequisites for BGP PA Output Interface Accounting

Before using the BGP Policy Accounting Output Interface Accounting feature, you must enable BGP and Cisco Express Forwarding or distributed CEF on the router.

Information About BGP PA Output Interface Accounting

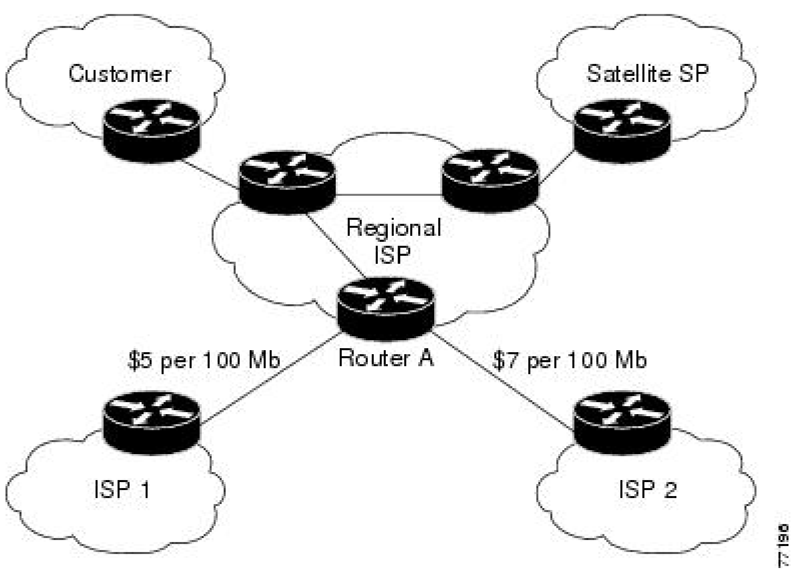
BGP PA Output Interface Accounting

Policy accounting using BGP measures and classifies IP traffic that is sent to, or received from, different peers. Originally, BGP PA was available on an input interface only. BGP PA output interface accounting introduces several extensions to enable BGP PA on an output interface and to include accounting based on a source address for both input and output traffic on an interface. Counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic.

Using the BGP **table-map** command, prefixes added to the routing table are classified by BGP attribute, autonomous system number, or autonomous system path. Packet and byte counters are incremented per input or output interface. A Cisco policy-based classifier maps the traffic into one of eight possible buckets that represent different traffic classes.

Using BGP PA, you can account for traffic according to its origin or the route it traverses. Service providers (SPs) can identify and account for all traffic by customer and can bill accordingly. In the figure below, BGP PA can be implemented in Router A to measure packet and byte volumes in autonomous system buckets. Customers are billed appropriately for traffic that is routed from a domestic, international, or satellite source.

Figure 51: Sample Topology for BGP Policy Accounting



BGP policy accounting using autonomous system numbers can be used to improve the design of network circuit peering and transit agreements between Internet service providers (ISPs).

Benefits of BGP PA Output Interface Accounting

Accounting for IP Traffic Differentially

BGP policy accounting classifies IP traffic by autonomous system number, autonomous system path, or community list string, and increments packet and byte counters. Policy accounting can also be based on the source address. Service providers can account for traffic and apply billing according to the origin of the traffic or the route that specific traffic traverses.

Efficient Network Circuit Peering and Transit Agreement Design

Implementing BGP policy accounting on an edge router can highlight potential design improvements for peering and transit agreements.

How to Conrigure BGP PA Output Interface Accounting

Specifying the Match Criteria for BGP PA

The first task in configuring BGP PA is to specify the criteria that must be matched. Community lists, autonomous system paths, or autonomous system numbers are examples of BGP attributes that can be specified and subsequently matched using a route map. Perform this task to specify the BGP attribute to use for BGP PA and to create the match criteria in a route map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip community-list** {*standard-list-number* | *expanded-list-number* [*regular-expression*] | {**standard** | **expanded**} *community-list-name*} {**permit** | **deny**} {*community-number* | *regular-expression*}
4. **route-map** *map-name* [**permit** | **deny**] [*sequence-number*]
5. **match community-list** *community-list-number* [**exact**]
6. **set traffic-index** *bucket-number*
7. **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	**ip community-list** {*standard-list-number* | *expanded-list-number* [*regular-expression*] | {**standard** | **expanded**} *community-list-name*} {**permit** | **deny**} {*community-number* | *regular-expression*} **Example:**	Creates a community list for BGP and controls access to it. • Repeat this step for each community to be specified.

	Command or Action	Purpose
	`Device(config)# ip community-list 30 permit 100:190`	
Step 4	**route-map** *map-name* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map set_bucket permit 10`	Enters route-map configuration mode and defines the conditions for policy routing. • The *map-name* argument identifies a route map. • The optional **permit** and **deny** keywords work with the match and set criteria to control how the packets are accounted for. • The optional *sequence-number* argument indicates the position that a new route map is to have in the list of route maps already configured with the same name.
Step 5	**match community-list** *community-list-number* [**exact**] **Example:** `Router(config-route-map)# match community-list 30`	Matches a BGP community.
Step 6	**set traffic-index** *bucket-number* **Example:** `Device(config-route-map)# set traffic-index 2`	Indicates where to output packets that pass a match clause of a route map for BGP policy accounting.
Step 7	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and returns to global configuration mode.

Classifying the IP Traffic and Enabling BGP PA

After a route map has been defined to specify match criteria, you must configure a way to classify the IP traffic before enabling BGP policy accounting.

Using the **table-map** command, BGP classifies each prefix that it adds to the routing table according to the match criteria. When the **bgp-policy accounting** command is configured on an interface, BGP policy accounting is enabled.

Perform this task to classify the IP traffic and enable BGP policy accounting.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **table-map** *route-map-name*
5. **network** *network-number* [**mask** *network-mask*]
6. **neighbor** *ip-address* **remote-as** *as-number*

7. **exit**
8. **interface** *type number*
9. **ip address** *ip-address mask*
10. **bgp-policy accounting** [**input** | **output**] [**source**]
11. **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`	Configures a BGP routing process and enters router configuration mode for the specified routing process. • The *as-number* argument identifies a BGP autonomous system number.
Step 4	**table-map** *route-map-name* **Example:** `Device(config-router)# table-map set_bucket`	Classifies BGP prefixes entered in the routing table.
Step 5	**network** *network-number* [**mask** *network-mask*] **Example:** `Device(config-router)# network 10.15.1.0 mask 255.255.255.0`	Specifies a network to be advertised by the BGP routing process.
Step 6	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `Device(config-router)# neighbor 10.14.1.1 remote-as 65100`	Specifies a BGP peer by adding an entry to the BGP routing table.
Step 7	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and returns to global configuration mode.
Step 8	**interface** *type number* **Example:**	Specifies the interface type and number and enters interface configuration mode.

	Command or Action	Purpose
	`Device(config)# interface POS 7/0`	• The *type* argument identifies the type of interface. • The *number* argument identifies the slot and port numbers of the interface. The space between the interface type and number is optional.
Step 9	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip-address 10.15.1.2 255.255.255.0`	Configures the interface with an IP address.
Step 10	**bgp-policy accounting** [**input** | **output**] [**source**] **Example:** `Device(config-if)# bgp-policy accounting input source`	Enables BGP policy accounting for the interface. • Use the optional **input** or **output** keyword to account for traffic either entering or leaving the router. By default, BGP policy accounting is based on traffic entering the router. • Use the optional **source** keyword to account for traffic based on source address.
Step 11	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

Verifying BGP Policy Accounting

Perform this task to verify that BGP policy accounting is operating.

SUMMARY STEPS

1. **show ip cef** [*network* [*mask*]] [**detail**]
2. **show ip bgp** [*network*] [*network-mask*] [**longer-prefixes**]
3. **show cef interface** [*type number*] **policy-statistics** [**input** | **output**]
4. **show cef interface** [*type number*] [**statistics**] [**detail**]

DETAILED STEPS

Step 1 **show ip cef** [*network* [*mask*]] [**detail**]

Enter the **show ip cef** command with the **detail** keyword to learn which accounting bucket is assigned to a specified prefix.

In this example, the output is displayed for the prefix 192.168.5.0. It shows that accounting bucket number 4 (traffic_index 4) is assigned to this prefix.

Example:

```
Device# show ip cef 192.168.5.0 detail

192.168.5.0/24, version 21, cached adjacency to POS7/2
0 packets, 0 bytes, traffic_index 4
  via 10.14.1.1, 0 dependencies, recursive
   next hop 10.14.1.1, POS7/2 via 10.14.1.0/30
   valid cached adjacency
```

Step 2 **show ip bgp** [*network*] [*network-mask*] [**longer-prefixes**]

Enter the **show ip bgp** command for the same prefix used in Step 1--192.168.5.0--to learn which community is assigned to this prefix.

In this example, the output is displayed for the prefix 192.168.5.0. It shows that the community of 100:197 is assigned to this prefix.

Example:

```
Device# show ip bgp 192.168.5.0

BGP routing table entry for 192.168.5.0/24, version 2
Paths: (1 available, best #1)
  Not advertised to any peer
  100
    10.14.1.1 from 10.14.1.1 (32.32.32.32)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Community: 100:197
```

Step 3 **show cef interface** [*type number*] **policy-statistics** [**input** | **output**]

Displays the per-interface traffic statistics.

In this example, the output shows the number of packets and bytes that have been assigned to each accounting bucket:

Example:

```
Device# show cef interface policy-statistics input

FastEthernet1/0/0 is up (if_number 6)
  Corresponding hwidb fast_if_number 6
  Corresponding hwidb firstsw->if_number 6
  BGP based Policy accounting on input is enabled
 Index         Packets           Bytes
     1            9999          999900
     2               0               0
     3               0               0
     4               0               0
     5               0               0
     6               0               0
     7               0               0
     8               0               0
     9               0               0
    10               0               0
    11               0               0
    12               0               0
    13               0               0
    14               0               0
    15               0               0
    16               0               0
    17               0               0
    18               0               0
    19               0               0
```

```
    20              0               0
    21              0               0
    22              0               0
    23              0               0
    24              0               0
    25              0               0
    26              0               0
    27              0               0
    28              0               0
    29              0               0
    30              0               0
    31              0               0
    32              0               0
    33              0               0
    34           1234          123400
    35              0               0
    36              0               0
    37              0               0
    38              0               0
    39              0               0
    40              0               0
    41              0               0
    42              0               0
    43              0               0
    44              0               0
    45           1000          100000
    46              0               0
    47              0               0
    48              0               0
    49              0               0
    50              0               0
    51              0               0
    52              0               0
    53              0               0
    54           5123         1198782
    55              0               0
    56              0               0
    57              0               0
    58              0               0
    59              0               0
    60              0               0
    61              0               0
    62              0               0
    63              0               0
    64              0               0
```

Step 4 **show cef interface** [*type number*] [**statistics**] [**detail**]

Displays the state of BGP policy accounting on a specified interface.

In this example, the output shows that BGP policy accounting has been configured to be based on input traffic at Fast Ethernet interface 1/0/0:

Example:

```
Device# show cef interface Fast Ethernet 1/0/0

FastEthernet1/0/0 is up (if_number 6)
  Corresponding hwidb fast_if_number 6
  Corresponding hwidb firstsw->if_number 6
  Internet address is 10.1.1.1/24
  ICMP redirects are always sent
  Per packet load-sharing is disabled
  IP unicast RPF check is disabled
```

```
Inbound access list is not set
Outbound access list is not set
IP policy routing is disabled
BGP based policy accounting on input is enabled
BGP based policy accounting on output is disabled
Hardware idb is FastEthernet1/0/0 (6)
Software idb is FastEthernet1/0/0 (6)
Fast switching type 1, interface type 18
IP Distributed CEF switching enabled
IP Feature Fast switching turbo vector
IP Feature CEF switching turbo vector
Input fast flags 0x100, Output fast flags 0x0, Flags 0x0
ifindex 7(7)
Slot 1 Slot unit 0 VC -1
Transmit limit accumulator 0xE8001A82 (0xE8001A82)
IP MTU 1500
```

Configuration Examples for BGP PA Output Interface Accounting

Example: Specifying the Match Criteria for BGP Policy Accounting

In the following example, BGP communities are specified in community lists, and a route map named set_bucket is configured to match each of the community lists to a specific accounting bucket using the **set traffic-index** command:

```
ip community-list 30 permit 100:190
ip community-list 40 permit 100:198
ip community-list 50 permit 100:197
ip community-list 60 permit 100:296
!
route-map set_bucket permit 10
match community 30
set traffic-index 2
!
route-map set_bucket permit 20
match community 40
set traffic-index 3
!
route-map set_bucket permit 30
match community 50
set traffic-index 4
!
route-map set_bucket permit 40
match community 60
set traffic-index 5
```

Example: Classifying the IP Traffic and Enabling BGP Policy Accounting

In the following example, BGP policy accounting is enabled on POS interface 7/0 and the **table-map** command is used to modify the bucket number when the IP routing table is updated with routes learned from BGP:

```
router bgp 65000
 table-map set_bucket
```

```
 network 10.15.1.0 mask 255.255.255.0
 neighbor 10.14.1.1 remote-as 65100
!
ip classless
ip bgp-community new-format
!
interface POS7/0
 ip address 10.15.1.2 255.255.255.0
 no ip directed-broadcast
 bgp-policy accounting
 no keepalive
 crc 32
 clock source internal
```

Additional References

The following sections provide references related to BGP policy accounting.

Related Documents

Related Topic	Document Title
BGP commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS IP Routing: BGP Command Reference
Switching commands: complete command syntax, command mode, defaults, usage guidelines, and examples	Cisco IOS IP Switching Command Reference
CEF and dCEF configuration information	*IP Switching Cisco Express Forwarding Configuration Guide*

Standards

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

MIBs

MIBs	MIBs Link
CISCO-BGP-POLICY-ACCOUNTING-MIB	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

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

Technical Assistance

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

Feature Information for BGP Policy Accounting Output Interface Accounting

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 42: Feature Information for BGP Policy Accounting Output Interface Accounting

Feature Name	Releases	Feature Information
BGP Policy Accounting Output Interface Accounting	12.0(9)S 12.0(17)ST 12.3(4)T Cisco IOS XE Release 2.1	BGP policy accounting (PA)measures and classifies IP traffic that is sent to, or received from, different peers. Policy accounting was previously available on an input interface only. The BGP Policy Accounting Output Interface Accounting feature introduces several extensions to enable BGP PA on an output interface and to include accounting based on a source address for both input and output traffic on an interface. Counters based on parameters such as community list, autonomous system number, or autonomous system path are assigned to identify the IP traffic. The following commands were introduced or modified: • **bgp-policy** • **set traffic-index** • **show cef interface** • **show cef interface policy statistics**

CHAPTER 30

BGP Cost Community

The BGP Cost Community feature introduces the cost extended community attribute. The cost community is a non-transitive extended community attribute that is passed to internal BGP (iBGP) and confederation peers but not to external BGP (eBGP) peers. The cost community feature allows you to customize the local route preference and influence the best path selection process by assigning cost values to specific routes.

In Cisco IOS Release 12.0(27)S, 12.3(8)T, 12.2(25)S, and later releases, support was introduced for mixed EIGRP MPLS VPN network topologies that contain VPN and backdoor links.

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 the BGP Cost Community Feature

This document assumes that BGP is configured in your network and that peering has been established.

Restrictions for the BGP Cost Community Feature

- The BGP Cost Community feature can be configured only within an autonomous system or confederation. The cost community is a non-transitive extended community that is passed to iBGP and confederation peers only and is not passed to eBGP peers.
- The BGP Cost Community feature must be supported on all routers in the autonomous system or confederation before cost community filtering is configured. The cost community should be applied consistently throughout the local autonomous system or confederation to avoid potential routing loops.
- Multiple cost community set clauses may be configured with the **set extcommunity cost** command in a single route map block or sequence. However, each set clause must be configured with a different ID value (0-255) for each point of insertion (POI). The ID value determines preference when all other attributes are equal. The lowest ID value is preferred.

Information About the BGP Cost Community Feature

BGP Cost Community Overview

The cost community is a nontransitive, extended community attribute that is passed to iBGP and confederation peers, but not to eBGP peers. The configuration of the BGP Cost Community feature allows you to customize the BGP best path selection process for a local autonomous system or confederation.

The cost community attribute is applied to internal routes by configuring the **set extcommunity cost** command in a route map. The cost community set clause is configured with a cost community ID number (0-255) and cost number (0-4294967295). The cost community ID number determines the preference for the path selection process. The path with the lowest cost community ID number is preferred.

Paths that are not specifically configured with the cost community attribute are assigned a default cost number value of 2147483647 (The midpoint between 0 and 4294967295) and evaluated by the best path selection process accordingly. In the case where two paths have been configured with the same cost community ID number, the path selection process will then prefer the path with the lowest cost number. The cost extended community attribute is propagated to iBGP peers when extended community exchange is enabled with the **neighbor send-community** command.

The following commands can be used to apply a route map that is configured with the cost community set clause:

- **aggregate-address**
- **neighbor default-originate route-map {in | out}**
- **neighbor route-map**
- **network route-map**
- **redistribute route-map**

How the BGP Cost Community Influences the Best Path Selection Process

The cost community attribute influences the BGP best path selection process at the point of insertion (POI). By default, the POI follows the IGP metric comparison. When BGP receives multiple paths to the same destination, it uses the best path selection process to determine which path is the best path. BGP automatically makes the decision and installs the best path into the routing table. The POI allows you to assign a preference to o a specific path when multiple equal cost paths are available. If the POI is not valid for local best path selection, the cost community attribute is silently ignored.

Multiple paths can be configured with the cost community attribute for the same POI. The path with the lowest cost community ID is considered first. In other words, all of the cost community paths for a specific POI are considered, starting with the one with the lowest cost community. Paths that do not contain the cost community (for the POI and community ID being evaluated) are assigned the default community cost value (2147483647). If the cost community values are equal, then cost community comparison proceeds to the next lowest community ID for this POI.

Note Paths that are not configured with the cost community attribute are considered by the best path selection process to have the default cost-value (half of the maximum value [4294967295] or 2147483647).

Applying the cost community attribute at the POI allows you to assign a value to a path originated or learned by a peer in any part of the local autonomous system or confederation. The cost community can be used as a “tie breaker” during the best path selection process. Multiple instances of the cost community can be configured for separate equal cost paths within the same autonomous system or confederation. For example, a lower cost community value can be applied to a specific exit path in a network with multiple equal cost exits points, and the specific exit path will be preferred by the BGP best path selection process. See the scenario described in the “Influencing Route Preference in a Multi-Exit IGP Network” section.

Cost Community Support for Aggregate Routes and Multipaths

Aggregate routes and multipaths are supported by the BGP Cost Community feature. The cost community attribute can be applied to either type of route. The cost community attribute is passed to the aggregate or multipath route from component routes that carry the cost community attribute. Only unique IDs are passed, and only the highest cost of any individual component route will be applied to the aggregate on a per-ID basis. If multiple component routes contain the same ID, the highest configured cost is applied to the route. For example, the following two component routes are configured with the cost community attribute via an inbound route map:

- 10.0.0.1 (POI=IGP, ID=1, Cost=100)
- 192.168.0.1 (POI=IGP, ID=1, Cost=200)

If these component routes are aggregated or configured as a multipath, the cost value 200 (POI=IGP, ID=1, Cost=200) will be advertised because it is the highest cost.

If one or more component routes does not carry the cost community attribute or if the component routes are configured with different IDs, then the default value (2147483647) will be advertised for the aggregate or multipath route. For example, the following three component routes are configured with the cost community attribute via an inbound route map. However, the component routes are configured with two different IDs.

- 10.0.0.1 (POI=IGP, ID=1, Cost=100)

- 172.16.0.1 (POI=IGP, ID=2, Cost=100)
- 192.168.0.1 (POI=IGP, ID=1, Cost=200)

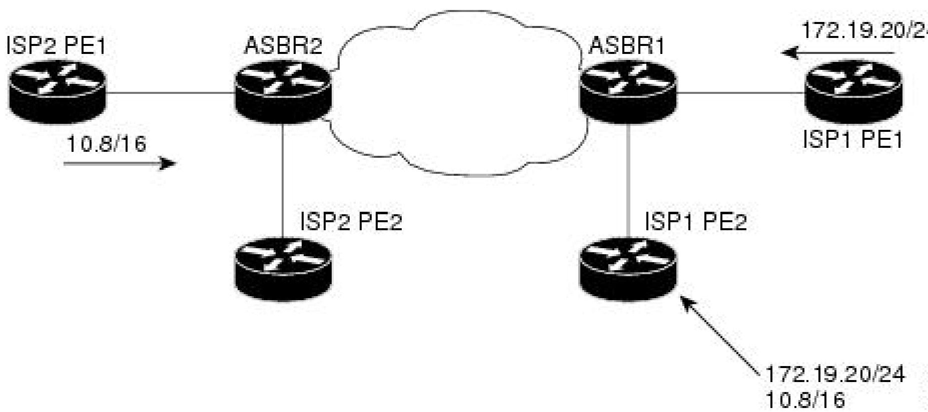
The single advertised path will include the aggregated cost communities as follows:

- {POI=IGP, ID=1, Cost=2147483647} {POI=IGP, ID=2, Cost=2147483647}

Influencing Route Preference in a Multi-Exit IGP Network

The figure below shows an Interior Gateway Protocol (IGP) network with two autonomous system boundary routers (ASBRs) on the edge. Each ASBR has an equal cost path to network 10.8/16.

Figure 52: Multi-Exit Point IGP Network



Both paths are considered to be equal by BGP. If multipath loadsharing is configured, both paths will be installed to the routing table and will be used to load balance traffic. If multipath load balancing is not configured, then BGP will select the path that was learned first as the best path and install this path to the routing table. This behavior may not be desirable under some conditions. For example, the path is learned from ISP1 PE2 first, but the link between ISP1 PE2 and ASBR1 is a low-speed link.

The configuration of the cost community attribute can be used to influence the BGP best path selection process by applying a lower cost community value to the path learned by ASBR2. For example, the following configuration is applied to ASBR2.

```
route-map ISP2_PE1 permit 10
 set extcommunity cost 1 1
 match ip address 13
!
ip access-list 13 permit 10.8.0.0 0.0.255.255
```

The above route map applies a cost community number value of 1 to the 10.8.0.0 route. By default, the path learned from ASBR1 will be assigned a cost community value of 2147483647. Because the path learned from ASBR2 has lower cost community value, this path will be preferred.

BGP Cost Community Support for EIGRP MPLS VPN PE-CE with Backdoor Links

Before EIGRP Site of Origin (SoO) BGP Cost Community support was introduced, BGP preferred locally sourced routes over routes learned from BGP peers. Back door links in an EIGRP MPLS VPN topology will be preferred by BGP if the back door link is learned first. (A back door link, or a route, is a connection that is configured outside of the VPN between a remote and main site. For example, a WAN leased line that connects a remote site to the corporate network).

The "pre-bestpath" point of insertion (POI) was introduced in the BGP Cost Community feature to support mixed EIGRP VPN network topologies that contain VPN and backdoor links. This POI is applied automatically to EIGRP routes that are redistributed into BGP. The "pre-best path" POI carries the EIGRP route type and metric. This POI influences the best path calculation process by influencing BGP to consider this POI before any other comparison step. No configuration is required. This feature is enabled automatically for EIGRP VPN sites when Cisco IOS Release 12.0(27)S is installed to a PE, CE, or back door router.

For information about configuring EIGRP MPLS VPNs, refer to the MPLS VPN Support for EIGRP Between Provider Edge and Customer Edge document in Cisco IOS Release 12.0(27)S.

For more information about the EIGRP MPLS VPN PE-CE Site of Origin (SoO) feature, refer to the EIGRP MPLS VPN PE-CE Site of Origin (SoO) feature documentation in Cisco IOS Release 12.0(27)S.

How to Configure the BGP Cost Community

Configuring the BGP Cost Community

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** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **ipv6** [**multicast** | **unicast**] | **vpnv4** [**unicast**]
6. **neighbor** *ip-address* **route-map** *map-name* {**in** | **out**}
7. **exit**
8. **route-map** *map-name* {**permit** | **deny**} [*sequence-number*]
9. **set extcommunity cost** [**igp**] *community-id cost-value*
10. **end**

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.

	Command or Action	Purpose
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 50000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.0.0.1 remote-as 101`	Establishes peering with the specified neighbor or peer-group.
Step 5	**address-family ipv4** [**mdt** | **multicast** | **tunnel** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **ipv6** [**multicast** | **unicast**] | **vpnv4** [**unicast**] **Example:** `Device(config-router)# address-family ipv4`	Places the router in address family configuration mode.
Step 6	**neighbor** *ip-address* **route-map** *map-name* {**in** | **out**} **Example:** `Device(config-router)# neighbor 10.0.0.1 route-map MAP-NAME in`	Applies an incoming or outgoing route map for the specified neighbor or peer-group.
Step 7	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode and enters global configuration mode.
Step 8	**route-map** *map-name* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map MAP-NAME permit 10`	Enters route map configuration mode to create or configure a route map.
Step 9	**set extcommunity cost** [**igp**] *community-id cost-value* **Example:** `Device(config-route-map)# set extcommunity cost 1 100`	Creates a set clause to apply the cost community attribute. • Multiple cost community set clauses can be configured in each route map block or sequence. Each cost community set clause must have a different ID (0-255). The cost community set clause with the lowest *cost-value* is preferred by the best path selection process when all other attributes are equal. • Paths that are not configured with the cost community attribute will be assigned the default *cost-value*, which is half of the maximum value (4294967295) or 2147483647.

	Command or Action	Purpose
Step 10	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.

Verifying the Configuration of the BGP Cost Community

BGP cost community configuration can be verified locally or for a specific neighbor. To verify the local configuration cost community, use the **show route-map** or **show running-config** command.

To verify that a specific neighbor carries the cost community, use the **show ip bgp** *ip-address* command. The output from these commands displays the POI (IGP is the default POI), the configured ID, and configured cost. For large cost community values, the output from these commands will also show, with + and - values, the difference between the configured cost and the default cost. See "Example: BGP Cost Community Verification" section for sample output.

Troubleshooting Tips

The **bgp bestpath cost-community ignore** command can be used to disable the evaluation of the cost community attribute to help isolate problems and troubleshoot issues that relate to BGP best path selection.

The **debug ip bgp updates** command can be used to print BGP update messages. The cost community extended community attribute will be displayed in the output of this command when received from a neighbor. A message will also be displayed if a non-transitive extended community if received from an external peer.

Configuration Examples for the BGP Cost Community Feature

Example: BGP Cost Community Configuration

The following example applies the cost community ID of 1 and cost community value of 100 to routes that are permitted by the route map. This configuration will cause the best path selection process to prefer this route over other equal-cost paths that were not permitted by this route map sequence.

```
Device(config)# router bgp 50000
Device(config-router)# neighbor 10.0.0.1 remote-as 50000
Device(config-router)# neighbor 10.0.0.1 update-source Loopback 0
Device(config-router)# address-family ipv4
Device(config-router-af)# neighbor 10.0.0.1 activate
Device(config-router-af)# neighbor 10.0.0.1 route-map COST1 in
Device(config-router-af)# neighbor 10.0.0.1 send-community both
Router(config-router-af)# exit
Device(config)# route-map COST1 permit 10
Device(config-route-map)# match ip-address 1
Device(config-route-map)# set extcommunity cost 1 100
```

Example: BGP Cost Community Verification

BGP cost community configuration can be verified locally or for a specific neighbor. To verify the local configuration cost community, use the **show route-map** or **show running-config** command. To verify that a specific neighbor carries the cost community, use the **show ip bgp** *ip-address* command.

The output of the **show route-map** command will display locally configured route-maps, match, set, continue clauses, and the status and configuration of the cost community attribute. The following sample output is similar to the output that will be displayed:

```
Device# show route-map

route-map COST1, permit, sequence 10
  Match clauses:
    as-path (as-path filter): 1
  Set clauses:
    extended community Cost:igp:1:100
  Policy routing matches: 0 packets, 0 bytes
route-map COST1, permit, sequence 20
  Match clauses:
    ip next-hop (access-lists): 2
  Set clauses:
    extended community Cost:igp:2:200
  Policy routing matches: 0 packets, 0 bytes
route-map COST1, permit, sequence 30
  Match clauses:
    interface FastEthernet0/0
    extcommunity (extcommunity-list filter):300
  Set clauses:
    extended community Cost:igp:3:300
  Policy routing matches: 0 packets, 0 bytes
```

The following sample output shows locally configured routes with large cost community values:

```
Device# show route-map

route-map set-cost, permit, sequence 10
  Match clauses:
  Set clauses:
    extended community RT:1:1 RT:2:2 RT:3:3 RT:4:4 RT:5:5 RT:6:6 RT:7:7
      RT:100:100 RT:200:200 RT:300:300 RT:400:400 RT:500:500 RT:600:600
      RT:700:700 additive
    extended community Cost:igp:1:4294967295 (default+2147483648)
      Cost:igp:2:200 Cost:igp:3:300 Cost:igp:4:400
      Cost:igp:5:2147483648 (default+1) Cost:igp:6:2147484648 (default+1001)
      Cost:igp:7:2147284648 (default-198999)
  Policy routing matches: 0 packets, 0 bytes
```

The output of the **show running config** command will display match, set, and continue clauses that are configured within a route-map. The following sample output is filtered to show only the relevant part of the running configuration:

```
Device# show running-config | begin route-map

route-map COST1 permit 20
 match ip next-hop 2
 set extcommunity cost igp 2 200
!
route-map COST1 permit 30
 match interface FastEthernet0/0
```

```
 match extcommunity 300
 set extcommunity cost igp 3 300
.
.
.
```

The output of the **show ip bgp** *ip-address* command can be used to verify if a specific neighbor carries a path that is configured with the cost community attribute. The cost community attribute information is displayed in the "Extended Community" field. The POI, the cost community ID, and the cost community number value are displayed. The following sample output shows that neighbor 172.16.1.2 carries a cost community with an ID of 1 and a cost of 100:

```
Device# show ip bgp 10.0.0.0

BGP routing table entry for 10.0.0.0/8, version 2
Paths: (1 available, best #1)
  Not advertised to any peer
  2 2 2
    172.16.1.2 from 172.16.1.2 (172.16.1.2)
      Origin IGP, metric 0, localpref 100, valid, external, best
      Extended Community: Cost:igp:1:100
```

If the specified neighbor is configured with the default cost community number value or if the default value is assigned automatically for cost community evaluation, "default" with + and - values will be displayed after the cost community number value in the output.

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
EIGRP MPLS VPN PE-CE Site of Origin (SoO) feature	"EIGRP MPLS VPN PE-CE Site of Origin (SoO)" module in the *IP Routing: EIGRP Configuration Guide*

Standards

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

MIBs

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

RFCs

RFCs	Title
draft-retana-bgp-custom-decision-00.txt	BGP Custom Decision Process

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 Cost Community

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 43: Feature Information for BGP Cost Community

Feature Name	Releases	Feature Information
BGP Cost Community	12.0(24)S 12.3(2)T 12.2(18)S 12.2(27)SBC 15.0(1)S	The BGP Cost Community feature introduces the cost extended community attribute. The cost community is a non-transitive extended community attribute that is passed to internal BGP (iBGP) and confederation peers but not to external BGP (eBGP) peers. The cost community feature allows you to customize the local route preference and influence the best path selection process by assigning cost values to specific routes. The following commands were introduced or modified: **bgp bestpath cost-community ignore**, **debug ip bgp updates**, and **set extcommunity cost**.

Feature Name	Releases	Feature Information
BGP Cost Community Support for EIGRP MPLS VPN PE-CE with Backdoor Links	12.0(27)S 12.3(8)T 12.2(25)S	Back door links in an EIGRP MPLS VPN topology will be preferred by BGP if the back door link is learned first. The "pre-bestpath" point of insertion (POI) was introduced in the BGP Cost Community feature to support mixed EIGRP VPN network topologies that contain VPN and backdoor links. This POI is applied automatically to EIGRP routes that are redistributed into BGP and the POI influences the best path calculation process by influencing BGP to consider this POI before any other comparison step. No configuration is required. This feature is enabled automatically for EIGRP VPN sites when Cisco IOS Release 12.0(27)S, 12.3(8)T, 12,2(25)S or later releases, is installed to a PE, CE, or back door router. No commands were introduced or modified.

CHAPTER 31

Regex Engine Performance Enhancement

The Regex Engine Performance Enhancement feature introduces a new regular expression engine that is designed to process complex regular expressions. This new regular expression engine does not replace the existing engine. The existing engine is preferred for simple regular expressions and is the default engine and in Cisco IOS software. Either engine can be selected from the command-line interface (CLI).

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 Regex Engine Performance Enhancement

The regular expression engine can be selected only under a Border Gateway Protocol (BGP) routing process in router configuration mode. So, the engine can be changed only after BGP has been enabled.

Information About Regex Engine Performance Enhancement

Regular Expression Overview

A regular expression is a pattern to match against an input string. You specify the pattern that a string must match when you compose a regular expression. Matching a string to the specified pattern is called "pattern matching." Pattern matching either succeeds or fails.

A regular expression can be a single-character pattern or a multiple-character pattern. That is, a regular expression can be a single character that matches the same single character in the input string or multiple characters that match the same multiple characters in the input string.

Default Regular Expression Engine

The default Cisco IOS regular expression engine uses a recursive algorithm. This engine is effective but uses more system resources as the complexity of regular expressions increase. The recursive algorithm works well for simple regular expressions, but is less efficient when processing very complex regular expressions because of the backtracking that is required by the default engine to process partial matches. In some cases, CPU watchdog timeouts and stack overflow traces have occurred because of the length of time that the default engine requires to process very complex regular expressions.

New Regular Expression Engine Selection

The Regex Engine Performance Enhancement feature introduces a deterministic processing time regular expression engine in Cisco IOS software. This new engine does not replace the default regular expression engine. The new engine employs an improved algorithm that eliminates excessive back tracking and greatly improves performance when processing complex regular expressions. When the new engine is enabled, complex regular expressions are evaluated more quickly, and CPU watchdog timeouts and stack overflow traces will not occur. However, the new regular expression engine takes longer to process simple regular expressions than the default engine.

We recommend that you use the new regular expression engine if you need to evaluate complex regular expressions or if you have observed problems related to evaluating regular expressions. We recommend that you use the default regular expression engine if you use only simple regular expressions. The new engine can be enabled by entering the **bgp regexp deterministic** command under a BGP routing process. The default regular expression engine can be reenabled by entering the **no** form of this command.

How to Change the Regular Expression Engine

Selecting the New Regular Expression Engine

We recommend that you use the new regular expression engine if you need to evaluate complex regular expressions or if you have observed problems related to evaluating regular expressions. We recommend that you use the default regular expression engine if you only use simple regular expressions.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** as-*number*
4. **bgp regexp deterministic**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** as-*number* **Example:** `Router(config)# router bgp 1`	Enters router configuration mode, and creates a BGP routing process.
Step 4	**bgp regexp deterministic** **Example:** `Router(config-router)# no bgp regexp deterministic`	Configures Cisco IOS to use a deterministic regular expression engine. • The default regular expression engine in Cisco IOS software is nondeterministic. • The default engine can be restored by entering the **no** form of this command.
Step 5	**exit** **Example:** `Router(config-router)# exit`	Exits router configuration mode, and enters global configuration mode.

Examples

The following example configures Cisco IOS software to use the default regular expression engine:

```
router bgp 1
 no bgp regexp deterministic
```

The following example configures Cisco IOS software to use the deterministic processing time regular expression engine:

```
router bgp 1
 bgp regexp deterministic
```

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
Regular Expressions	"Regular Expressions" appendix of the *Cisco IOS Terminal Services Configuration Guide*

Standards

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

MIBs

MIBs	MIBs Link
—	To obtain lists of supported MIBs by platform and Cisco IOS release, and to download MIB modules, go to the Cisco MIB website on Cisco.com at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

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

Technical Assistance

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

Feature Information for Regex Performance Enhancement

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 44: Feature Information for Regex Performance Enhancement

Feature Name	Releases	Feature Information
Regex Performance Enhancement	12.0(26)S 12.3(4)T 12.2(22)S Cisco IOS XE 3.1.0SG	The Regex Engine Performance Enhancement feature introduces a new regular expression engine that is designed to process complex regular expressions. This new regular expression engine does not replace the existing engine. The existing engine is preferred for simple regular expressions and is the default engine and in Cisco IOS software. Either engine can be selected from the command-line interface (CLI). The following command was introduced: **bgp regexp deterministic**.

CHAPTER 32

BGP Support for IP Prefix Import from Global Table into a VRF Table

The BGP Support for IP Prefix Import from Global Table into a VRF Table feature introduces the capability to import IPv4 unicast prefixes from the global routing table into a Virtual Private Network (VPN) routing/forwarding (VRF) instance table using an import 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.

Prerequisites for BGP Support for IP Prefix Import from Global Table into a VRF Table

- Border Gateway Protocol (BGP) peering sessions are established.
- CEF or dCEF (for distributed platforms) is enabled on all participating routers.

Restrictions for BGP Support for IP Prefix Import from Global Table into a VRF Table

- Only IPv4 unicast and multicast prefixes can be imported into a VRF with this feature.
- IPv4 prefixes imported into a VRF using this feature cannot be imported into a VPNv4 VRF.
- The global prefixes should be in the BGP table, so that this feature can import them into the BGP VRF table.
- IPv4 prefixes imported into a VRF using this feature cannot be imported into a second VPNv4 VRF.

Information About BGP Support for IP Prefix Import from Global Table into a VRF Table

Importing IPv4 Prefixes into a VRF

The BGP Support for IP Prefix Import from Global Table into a VRF Table feature introduces the capability to import IPv4 unicast prefixes from the global routing table into a Virtual Private Network (VPN) routing/forwarding instance (VRF) table using an import route map. This feature extends the functionality of VRF import-map configuration to allow IPv4 prefixes to be imported into a VRF based on a standard community. Both IPv4 unicast and multicast prefixes are supported. No Multiprotocol Label Switching (MPLS) or route target (import/export) configuration is required.

IP prefixes are defined as match criteria for the import map through standard Cisco filtering mechanisms. For example, an IP access-list, an IP prefix-list, or an IP as-path filter is created to define an IP prefix or IP prefix range, and then the prefix or prefixes are processed through a match clause in a route map. Prefixes that pass through the route map are imported into the specified VRF per the import map configuration.

Black Hole Routing

The BGP Support for IP Prefix Import from Global Table into a VRF Table feature can be configured to support Black Hole Routing (BHR). BHR is a method that allows the administrator to block undesirable traffic, such as traffic from illegal sources or traffic generated by a Denial of Service (DoS) attack, by dynamically routing the traffic to a dead interface or to a host designed to collect information for investigation, mitigating the impact of the attack on the network. Prefixes are looked up, and packets that come from unauthorized sources are blackholed by the ASIC at line rate.

Classifying Global Traffic

The BGP Support for IP Prefix Import from Global Table into a VRF Table feature can be used to classify global IP traffic based on physical location or class of service. Traffic is classified based on administration policy and then imported into different VRFs. On a college campus, for example, network traffic could be divided into an academic network and residence network traffic, a student network and faculty network, or a dedicated network for multicast traffic. After the traffic is divided along administration policy, routing decisions can be configured with the MPLS VPN--VRF Selection Using Policy Based Routing feature or the MPLS VPN--VRF Selection Based on Source IP Address feature.

Unicast Reverse Path Forwarding

Unicast Reverse Path Forwarding (Unicast RPF) can be optionally configured with the BGP Support for IP Prefix Import from Global Table into a VRF Table feature. Unicast RPF is used to verify that the source address is in the Forwarding Information Base (FIB). The **ip verify unicast vrf** command is configured in interface configuration mode and is enabled for each VRF. This command has **permit** and **deny**keywords that are used to determine if the traffic is forwarded or dropped after Unicast RPF verification.

How to Import IP Prefixes from Global Table into a VRF Table

Defining IPv4 IP Prefixes to Import

IPv4 unicast or multicast prefixes are defined as match criteria for the import route map using standard Cisco filtering mechanisms. This task uses an IP access-list and an IP prefix-list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**]
4. **ip prefix-list** *prefix-list-name* [**seq** *seq-value*] {**deny** *network*/*length* | **permit** *network*/*length*} [**ge** *ge-value*] [**le** *le-value*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] [**log**] **Example:** `Device(config)# access-list 50 permit 10.1.1.0 0.0.0.255`	Creates an access list and defines a range of IP prefixes to import into the VRF table. • The example creates a standard access list numbered 50. This filter will permit traffic from any host with an IP address in the 10.1.1.0/24 subnet.

	Command or Action	Purpose
Step 4	**ip prefix-list** *prefix-list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} [**ge** *ge-value*] [**le** *le-value*] **Example:** `Device(config)# ip prefix-list COLORADO permit 10.24.240.0/22`	Creates a prefix list and defines a range of IP prefixes to import into the VRF table. • The example creates an IP prefix list named COLORADO. This filter will permit traffic from any host with an IP address in the 10.24.240.0/22 subnet.

Creating the VRF and the Import Route Map

The IP prefixes that are defined for import are then processed through a match clause in a route map. IP prefixes that pass through the route map are imported into the VRF. A maximum of 5 VRFs per router can be configured to import IPv4 prefixes from the global routing table. By default, a maximum of 1000 prefixes per VRF can be imported. You can change the limit to be from 1 to 2,147,483,647 prefixes for each VRF. We recommend that you use caution if you increase the prefix import limit above 1000. Configuring the router to import too many prefixes can interrupt normal router operation.

No MPLS or route target (import/export) configuration is required.

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

The following syslog message is introduced by the BGP Support for IP Prefix Import from Global Table into a VRF Table feature. It will be displayed when more prefixes are available for import than the user-defined limit:

```
00:00:33: %BGP-3-AFIMPORT_EXCEED: IPv4 Multicast prefixes imported to multicast vrf exceed
 the limit 2
```

You can either increase the prefix limit or fine-tune the import route map filter to reduce the number of candidate routes.

Note

- Only IPv4 unicast and multicast prefixes can be imported into a VRF with this feature.
- A maximum of five VRF instances per router can be created to import IPv4 prefixes from the global routing table.
- IPv4 prefixes imported into a VRF using this feature cannot be imported into a VPNv4 VRF.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*
5. **import ipv4** {**unicast** | **multicast**} [*prefix-limit*] **map** *route-map*
6. **exit**

7. **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*]
8. **match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]}
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Router(config)# ip vrf GREEN`	Creates a VRF routing table and specifies the VRF name (or tag). • The **ip vrf** *vrf-name* command creates a VRF routing table and a CEF table, and both are named using the *vrf-name* argument. Associated with these tables is the default route distinguisher value.
Step 4	**rd** *route-distinguisher* **Example:** `Router(config-vrf)# rd 100:10`	Creates routing and forwarding tables for the VRF instance. • There are two formats for configuring the route distinguisher argument. It can be configured in the as-number:network number (ASN:nn) format, as shown in the example, or it can be configured in the IP address:network number format (IP-address:nn).
Step 5	**import ipv4** {**unicast** | **multicast**} [*prefix-limit*] **map** *route-map* **Example:** `Router(config-vrf)# import ipv4 unicast 1000 map UNICAST`	Imports IPv4 prefixes from the global routing table to a VRF table, filtered by the specified route map. • Unicast or multicast prefixes are specified. • Up to a 1000 prefixes will be imported by default. The *prefix-limit* argument is used to specify a limit from 1 to 2,147,483,647 prefixes. • The example references a route map that will import up to 1000 unicast prefixes that pass through the route map.
Step 6	**exit** **Example:** `Router(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.

	Command or Action	Purpose
Step 7	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Router(config)# route-map UNICAST permit 10`	Defines the conditions for redistributing routes from one routing protocol into another, or enables policy routing. • The route map name must match the route map specified in Step 5. • The example creates a route map named UNICAST.
Step 8	**match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]} **Example:** `Router(config-route-map)# match ip address 50`	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, and performs policy routing on matched packets. • Both IP access lists and IP prefix lists are supported. • The example configures the route map to use standard access list 50 to define match criteria.
Step 9	**end** **Example:** `Router(config-route-map)# end`	Exits route-map configuration mode and returns to privileged EXEC mode.

Filtering on the Ingress Interface

The BGP Support for IP Prefix Import from Global Table into a VRF Table feature can be configured globally or on a per-interface basis. We recommend that you apply it to ingress interfaces to maximize performance.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **ip policy route-map** *map-tag*
5. **ip verify unicast vrf** *vrf-name* {**deny** | **permit**}
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**interface** *type number* [*name-tag*] **Example:** `Router(config)# interface Ethernet0/0`	Configures an interface and enters interface configuration mode.
Step 4	**ip policy route-map** *map-tag* **Example:** `Router(config-if)# ip policy route-map UNICAST`	Identifies a route map to use for policy routing on an interface. • The example attaches the route map named UNICAST to the interface.
Step 5	**ip verify unicast vrf** *vrf-name* {**deny** | **permit**} **Example:** `Router(config-if)# ip verify unicast vrf GREEN permit`	(Optional) Enables Unicast Reverse Path Forwarding verification for the specified VRF. • The example enables verification for the VRF named GREEN. Traffic that passes verification will be forwarded.
Step 6	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying Global IP Prefix Import

Perform the steps in this task to display information about the VRFs that are configured with the BGP Support for IP Prefix Import from Global Table into a VRF Table feature and to verify that global IP prefixes are imported into the specified VRF table.

SUMMARY STEPS

1. **enable**
2. **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*}
3. **show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*]

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device# enable
```

Step 2 **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*}

Displays VPN address information from the BGP table. The output displays the import route map, the traffic type (unicast or multicast), the default or user-defined prefix import limit, the actual number of prefixes that are imported, and individual import prefix entries.

Example:

```
Device# show ip bgp vpnv4 all

BGP table version is 15, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:1 (default for vrf academic)
Import Map: ACADEMIC, Address-Family: IPv4 Unicast, Pfx Count/Limit: 6/1000
*> 10.50.1.0/24     172.17.2.2                             0 2 3 ?
*> 10.50.2.0/24     172.17.2.2                             0 2 3 ?
*> 10.50.3.0/24     172.17.2.2                             0 2 3 ?
*> 10.60.1.0/24     172.17.2.2                             0 2 3 ?
*> 10.60.2.0/24     172.17.2.2                             0 2 3 ?
*> 10.60.3.0/24     172.17.2.2                             0 2 3 ?
Route Distinguisher: 200:1 (default for vrf residence)
Import Map: RESIDENCE, Address-Family: IPv4 Unicast, Pfx Count/Limit: 3/1000
*> 10.30.1.0/24     172.17.2.2                 0           0 2 i
*> 10.30.2.0/24     172.17.2.2                 0           0 2 i
*> 10.30.3.0/24     172.17.2.2                 0           0 2 i
Route Distinguisher: 300:1 (default for vrf BLACKHOLE)
Import Map: BLACKHOLE, Address-Family: IPv4 Unicast, Pfx Count/Limit: 3/1000
*> 10.40.1.0/24     172.17.2.2                 0           0 2 i
*> 10.40.2.0/24     172.17.2.2                 0           0 2 i
*> 10.40.3.0/24     172.17.2.2                 0           0 2 i
Route Distinguisher: 400:1 (default for vrf multicast)
Import Map: MCAST, Address-Family: IPv4 Multicast, Pfx Count/Limit: 2/2
*> 10.70.1.0/24     172.17.2.2                 0           0 2 i
*> 10.70.2.0/24     172.17.2.2                 0           0 2 i
```

Step 3 **show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*]

Displays defined VRFs and their associated interfaces. The output displays the import route map, the traffic type (unicast or multicast), and the default or user-defined prefix import limit. The following example output shows that the import route map named UNICAST is importing IPv4 unicast prefixes and that the prefix import limit is 1000.

Example:

```
Device# show ip vrf detail

VRF academic; default RD 100:10; default VPNID <not set>
VRF Table ID = 1
  No interfaces
  Connected addresses are not in global routing table
  Export VPN route-target communities
    RT:100:10
  Import VPN route-target communities
    RT:100:10
  Import route-map for ipv4 unicast: UNICAST (prefix limit: 1000)
  No export route-map
```

Configuration Examples for BGP Support for IP Prefix Import from Global Table into a VRF Table

Example: Importing IP Prefixes from Global Table into a VRF Table

The following example imports unicast prefixes into the VRF named *green* by using an IP prefix list and a route map:

This example starts in global configuration mode:

```
!
ip prefix-list COLORADO seq 5 permit 10.131.64.0/19
ip prefix-list COLORADO seq 10 permit 172.31.2.0/30
ip prefix-list COLORADO seq 15 permit 172.31.1.1/32
!
ip vrf green
 rd 200:1
 import ipv4 unicast map UNICAST
 route-target export 200:10
 route-target import 200:10
!
 exit
!
route-map UNICAST permit 10
 match ip address prefix-list COLORADO
!
 exit
```

Example: Verifying IP Prefix Import to a VRF Table

The **show ip vrf** command or the **show ip bgp vpnv4** command can be used to verify that prefixes are imported from the global routing table to the VRF table.

The following sample output shows that the import route map named UNICAST is importing IPv4 unicast prefixes and the prefix import limit is 1000:

```
Device# show ip vrf detail

VRF green; default RD 200:1; default VPNID <not set>
  Interfaces:
    Se2/0
VRF Table ID = 1
  Export VPN route-target communities
    RT:200:10
  Import VPN route-target communities
    RT:200:10
  Import route-map for ipv4 unicast: UNICAST (prefix limit: 1000)
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
VRF red; default RD 200:2; default VPNID <not set>
  Interfaces:
    Se3/0
VRF Table ID = 2
```

```
  Export VPN route-target communities
    RT:200:20
  Import VPN route-target communities
    RT:200:20
  No import route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
```

The following sample output displays the import route map names, the prefix import limit and the actual number of imported prefixes, and the individual import entries:

```
Device# show ip bgp vpnv4 all

BGP table version is 18, local router ID is 10.131.127.252
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 200:1 (default for vrf green)
Import Map: UNICAST, Address-Family: IPv4 Unicast, Pfx Count/Limit: 1/1000
*>i10.131.64.0/19   10.131.95.252            0    100      0 i
*> 172.16.1.1/32    172.16.2.1               0         32768 i
*> 172.16.2.0/30    0.0.0.0                  0         32768 i
*>i172.31.1.1/32    10.131.95.252            0    100      0 i
*>i172.31.2.0/30    10.131.95.252            0    100      0 i
Route Distinguisher: 200:2 (default for vrf red)
*> 172.16.1.1/32    172.16.2.1               0         32768 i
*> 172.16.2.0/30    0.0.0.0                  0         32768 i
*>i172.31.1.1/32    10.131.95.252            0    100      0 i
*>i172.31.2.0/30    10.131.95.252            0    100      0 i
```

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
MPLS Layer 3 VPN configuration tasks	"Configuring MPLS Layer 3 VPNs" module in the *MPLS: Layer 3 VPNs Configuration Guide*
VRF selection using policy based routing	"MPLS VPN VRF Selection Using Policy-Based Routing" module in the *MPLS: Layer 3 VPNs Configuration Guide*

Standards

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

MIBs

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

RFCs

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

Technical Assistance

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

Feature Information for BGP Support for IP Prefix Import from Global Table into a VRF Table

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

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

Table 45: Feature Information for BGP Support for IP Prefix Import from Global Table into a VRF Table

Feature Name	Releases	Feature Information
BGP Support for IP Prefix Import from Global Table into a VRF Table	12.0(29)S 12.2(25)S 12.2(27)SBC 12.2(33)SRA 12.2(33)SXH 12.3(14)T 15.0(1)S	The BGP Support for IP Prefix Import from Global Table into a VRF Table feature introduces the capability to import IPv4 unicast prefixes from the global routing table into a Virtual Private Network (VPN) routing/forwarding (VRF) instance table using an import route map. The following commands were introduced or modified by this feature: **debug ip bgp import**, **import ipv4**, and **ip verify unicast vrf**.

CHAPTER 33

BGP per Neighbor SoO Configuration

The BGP per Neighbor SoO Configuration feature simplifies the configuration of the site-of-origin (SoO) value. In Cisco IOS Release 12.4(9)T, 12.2(33)SRA, 12.2(31)SB2, and previous releases, the SoO value is configured using an inbound route map that sets the SoO value during the update process. Per neighbor SoO configuration introduces two new commands that can be configured in submodes under router configuration mode to set the SoO value. In Cisco IOS Release 12.4(24)T, support was added for 4-byte autonomous system numbers in asdot format only.

Finding Feature Information

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

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

Prerequisites for BGP per Neighbor SoO Configuration

This feature assumes that a Border Gateway Protocol (BGP) network is configured and that Cisco Express Forwarding is enabled in your network.

Restrictions for BGP per Neighbor SoO Configuration

A BGP neighbor or peer policy template-based SoO configuration takes precedence over the SoO value configured in an inbound route map.

Information About Configuring BGP per Neighbor SoO

Site of Origin BGP Community Attribute

The site-of-origin (SoO) extended community is a BGP extended community attribute that is used to identify routes that have originated from a site so that the readvertisement of that prefix back to the source site can be prevented. The SoO extended community uniquely identifies the site from which a router has learned a route. BGP can use the SoO value associated with a route to prevent routing loops.

Route Distinguisher

A route distinguisher (RD) creates routing and forwarding tables and specifies the default route distinguisher for a VPN. The RD is added to the beginning of an IPv4 prefix to change it into a globally unique VPN-IPv4 prefix. An RD can be composed in one of two ways: with an autonomous system number and an arbitrary number or with an IP address and an arbitrary number.

You can enter an RD in either of these formats:

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

45000:3

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

192.168.10.15:1

BGP per Neighbor Site of Origin Configuration

There are three ways to configure an SoO value for a BGP neighbor:

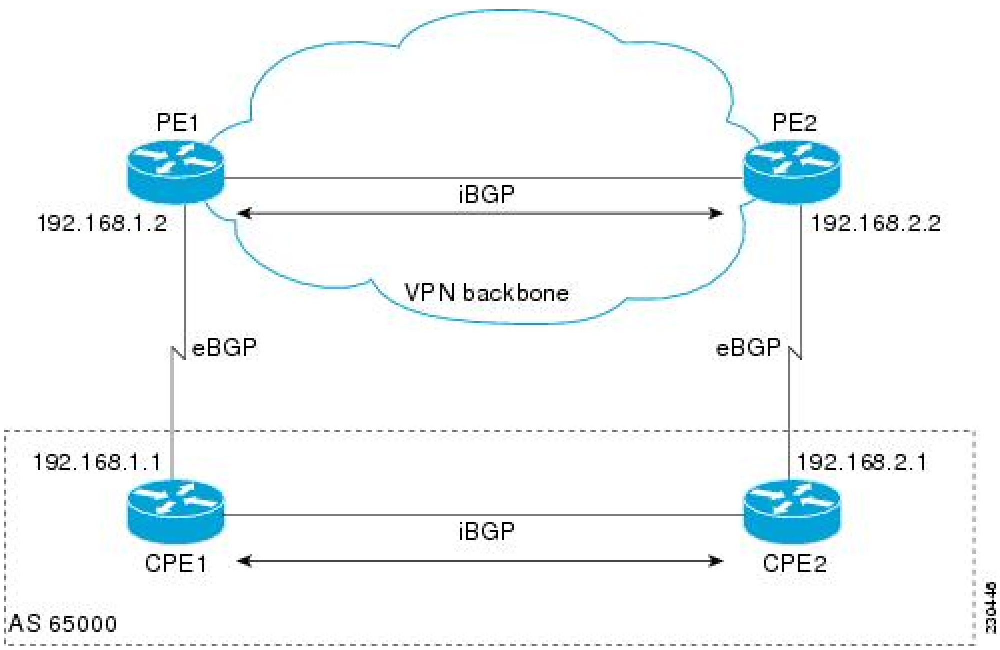
- BGP peer policy template--A peer policy template is created, and an SoO value is configured as part of the peer policy. Under address family IPv4 VRF, a neighbor is identified and is configured to inherit the peer policy that contains the SoO value.
- BGP **neighbor** command--Under address family IPv4 VRF, a neighbor is identified, and an SoO value is configured for the neighbor.
- BGP peer group--Under address family IPv4 VRF, a BGP peer group is configured, an SoO value is configured for the peer group, a neighbor is identified, and the neighbor is configured as a member of the peer group.

Note A BGP neighbor or peer policy template-based SoO configuration takes precedence over the SoO value configured in an inbound route map.

The configuration of SoO values for BGP neighbors is performed on a provider edge (PE) router, which is the VPN entry point. When SoO is enabled, the PE router forwards prefixes to the customer premises equipment (CPE) only when the SoO tag of the prefix does not match the SoO tag configured for the CPE.

For example, in the figure below, an SoO tag is set as 65000:1 for the customer site that includes routers CPE1 and CPE2 with an autonomous system number of 65000. When CPE1 sends prefixes to PE1, PE1 tags the prefixes with 65000:1, which is the SoO tag for CPE1 and CPE2. When PE1 sends the tagged prefixes to PE2, PE2 performs a match against the SoO tag from CPE2. Any prefixes with the tag value of 65000:1 are not sent to CPE2 because the SoO tag matches the SoO tag of CPE2, and a routing loop is avoided.

Figure 53: Network Diagram for SoO Example



Benefits of BGP per Neighbor Site of Origin

In releases prior to the introduction of this feature, the SoO extended community attribute is configured using an inbound route map that sets the SoO value during the update process. With the introduction of the BGP per Neighbor Site of Origin feature, two new commands configured in submodes under router configuration mode simplify the SoO value configuration.

How to Configure BGP per Neighbor SoO

Enabling Cisco Express Forwarding and Configuring VRF Instances

Perform this task on both of the PE routers in the figure above to configure Virtual Routing and Forwarding (VRF) instances to be used with the per-VRF assignment tasks. In this task, Cisco Express Forwarding is enabled, and a VRF instance named SOO_VRF is created. To make the VRF functional, a route distinguisher is created, and the VRF is associated with an interface. When the route distinguisher is created, the routing and forwarding tables are created for the VRF instance named SOO_VRF. After associating the VRF with an interface, the interface is configured with an IP address.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip cef**
4. **ip vrf** *vrf-name*
5. **rd** *route-distinguisher*
6. **route-target** {**export** | **both**} *route-target-ext-community*
7. **route-target** {**import** | **both**} *route-target-ext-community*
8. **exit**
9. **interface** *type number*
10. **ip vrf forwarding** *vrf-name* [**downstream** *vrf-name2*]
11. **ip address** *ip-address mask* [**secondary**]
12. **end**
13. **show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*] [*output-modifiers*]

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 cef** **Example:** `Device(config)# ip cef`	Enables Cisco Express Forwarding on the route processor.

	Command or Action	Purpose
Step 4	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf SOO_VRF`	Defines a VRF instance and enters VRF configuration mode.

Step 5	**rd** *route-distinguisher*
	Example:
	`Device(config-vrf)# rd 1:1`
		Creates routing and forwarding tables for a VRF and specifies the default RD for a VPN.
		• Use the *route-distinguisher* argument to specify the default RD for a VPN. There are two formats that you can use to specify an RD:
		  • A 16-bit autonomous system number, a colon, and a 32-bit number, for example: 65000:3
		  • A 32-bit IP address, a colon, and a 16-bit number, for example: 192.168.1.2:51
		• In this example, the RD uses an autonomous system number with the number 1 after the colon.

Step 6	**route-target** {**export** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 1:1`	Creates a route-target extended community for a VRF. • Use the **export** keyword to export routing information to the target VPN extended community. • Use the **both** keyword to both import routing information from, and export routing information to, the target VPN extended community. • Use the *route-target-ext-community* argument to specify the VPN extended community. **Note** Only the syntax applicable to this step is displayed. For a different use of this syntax, see Step 7.
Step 7	**route-target** {**import** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 1:1`	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 **both** keyword to both import routing information from, and export routing information to, the target VPN extended community. • Use the *route-target-ext-community* argument to specify the VPN extended community.
Step 8	**exit** **Example:**	Exits VRF configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Device(config-vrf)# exit`	
Step 9	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/0`	Configures an interface type and enters interface configuration mode.
Step 10	**ip vrf forwarding** *vrf-name* [**downstream** *vrf-name2*] **Example:** `Device(config-if)# ip vrf forwarding SOO_VRF`	Associates a VRF with an interface or subinterface. • In this example, the VRF named SOO_VRF is associated with Gigabit Ethernet interface 1/0/0. **Note** Executing this command on an interface removes the IP address, so the IP address should be reconfigured.
Step 11	**ip address** *ip-address mask* [**secondary**] **Example:** `Device(config-if)# ip address 192.168.1.2 255.255.255.0`	Configures an IP address. • In this example, Gigabit Ethernet interface 1/0/0 is configured with an IP address of 192.168.1.2.
Step 12	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 13	**show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*] [*output-modifiers*] **Example:** `Device# show ip vrf`	Displays the configured VRFs. • Use this command to verify the configuration of this task.

Examples

The following output of the **show ip vrf** command displays the VRF named SOO_VRF configured in this task.

```
Device# show ip vrf

Name                              Default RD            Interfaces
SOO_VRF                                  1:1                GE1/0/0
```

Configuring a per Neighbor SoO Value Using a BGP Peer Policy Template

Perform this task on router PE1 in the figure above to configure an SoO value for a BGP neighbor at the router CPE1 in the figure above using a peer policy template. In this task, a peer policy template is created, and the

SoO value is configured for the peer policy. Under address family IPv4 VRF, a neighbor is identified and is configured to inherit the peer policy that contains the SoO value.

For a configuration example involving 4-byte autonomous system numbers, see the “Example: Configuring a per Neighbor SoO Value with a 4-Byte AS Number Using a BGP Peer Policy Template” section.

Note If a BGP peer inherits from several peer policy templates that specify different SoO values, the SoO value in the last template applied takes precedence and is applied to the peer. However, direct configuration of the SoO value on the BGP neighbor overrides any inherited template configurations of the SoO value.

BGP Peer Policy Templates

Peer policy templates are used to configure BGP policy commands that are configured for neighbors that belong to specific address families. Peer policy templates are configured once and then applied to many neighbors through the direct application of a peer policy template or through inheritance from peer policy templates. The configuration of peer policy templates simplifies the configuration of BGP policy commands that are applied to all neighbors within an autonomous system.

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

The configuration of peer policy templates simplifies and improves the flexibility of BGP configuration. A specific policy can be configured once and referenced many times. Because a peer policy supports up to eight levels of inheritance, very specific and very complex BGP policies can be created.

For more details about BGP peer policy templates, see the "Configuring a Basic BGP Network" module.

Before you begin

This task assumes that the task described in the "Verifying CEF and Configuring VRF Instances" section has been performed.

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-policy** *policy-template-name*
5. **soo** *extended-community-value*
6. **exit-peer-policy**
7. **address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*]
8. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
9. **neighbor** *ip-address* **activate**
10. **neighbor** *ip-address* **inherit peer-policy** *policy-template-name*

11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 50000`	Enters router configuration mode for the specified routing process.
Step 4	**template peer-policy** *policy-template-name* **Example:** `Router(config-router)# template peer-policy SOO_POLICY`	Creates a peer policy template and enters policy-template configuration mode.

Step 5	**soo** *extended-community-value*
	Example:
	`Router(config-router-ptmp)# soo 65000:1`
		Sets the SoO value for a BGP peer policy template.
		• Use the *extended-community-value* argument to specify the VPN extended community value. The value takes one of the following formats:
		  • A 16-bit autonomous system number, a colon, and a 32-bit number, for example: 45000:3
		  • A 32-bit IP address, a colon, and a 16-bit number, for example: 192.168.10.2:51
		• In this example, the SoO value is set at 65000:1.

Step 6	**exit-peer-policy** **Example:** `Router(config-router-pmtp)# exit-peer-policy`	Exits policy-template configuration mode and returns to router configuration mode.
Step 7	**address-family ipv4** [**unicast** | **multicast**| **vrf** *vrf-name*] **Example:** `Router(config-router)# address-family ipv4 vrf SOO_VRF`	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 router is placed in configuration mode for the IPv4 unicast address

	Command or Action	Purpose
		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 8	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.1.1 remote-as 65000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 9	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local router.
Step 10	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Router(config-router-af)# neighbor 192.168.1.1 inherit peer-policy SOO_POLICY`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration. • In this example, the router is configured to send the peer policy template named SOO_POLICY to the 192.168.1.1 neighbor to inherit. If another peer policy template is indirectly inherited from SOO_POLICY, the indirectly inherited configuration will also be applied. Up to seven additional peer policy templates can be indirectly inherited from SOO_POLICY.
Step 11	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuring a per Neighbor SoO Value Using a BGP neighbor Command

Perform this task on router PE2 in the figure above to configure an SoO value for the BGP neighbor at router CPE2 in the figure above using a **neighbor** command. For the IPv4 VRF address family, a neighbor is identified, and an SoO value is configured for the neighbor.

Direct configuration of the SoO value on a BGP neighbor overrides any inherited peer policy template configurations of the SoO value.

Before you begin

This task assumes that the task described in the "Verifying CEF and Configuring VRF Instances" section has been performed with appropriate changes to interfaces and IP addresses.

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* | *peer-group-name*} **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **activate**
7. **neighbor** {*ip-address* | *peer-group-name*} **soo** *extended-community-value*
8. **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 50000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 vrf SOO_VRF`	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 router is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • 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.

	Command or Action	Purpose
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.2.1 remote-as 65000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 6	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.2.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local router. • In this example, the external BGP peer at 192.168.2.1 is activated. **Note** If a peer group has been configured in Step 5 , do not use this step because BGP peer groups are activated when any parameter is configured. For example, a BGP peer group is activated when an SoO value is configured using the **neighbor soo** command in Step 7.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **soo** *extended-community-value* **Example:** `Device(config-router-af)# neighbor 192.168.2.1 soo 65000:1`	Sets the site-of-origin (SoO) value for a BGP neighbor or peer group. • In this example, the neighbor at 192.168.2.1 is configured with an SoO value of 65000:1.
Step 8	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuring a per Neighbor SoO Value Using a BGP Peer Group

Perform this task on router PE1 in the figure above to configure an SoO value for the BGP neighbor at router CPE1 in the figure above using a **neighbor** command with a BGP peer group. Under address family IPv4 VRF, a BGP peer group is created and an SoO value is configured using a BGP **neighbor** command, and a neighbor is then identified and added as a peer group member. A BGP peer group member inherits the configuration associated with a peer group, which in this example, includes the SoO value.

Direct configuration of the SoO value on a BGP neighbor overrides any inherited peer group configurations of the SoO value.

Before you begin

This task assumes that the task described in “Enabling Cisco Express Forwarding and Configuring VRF Instances” has been performed.

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. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
5. **neighbor** *peer-group-name* **peer-group**
6. **neighbor** {*ip-address* | *peer-group-name*} **soo** *extended-community-value*
7. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
8. **neighbor** *ip-address* **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 50000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 vrf SOO_VRF`	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 router is placed in configuration mode for the IPv4 unicast address family if the **unicast** keyword is not specified with the **address-family ipv4** command. • Use the **multicast** keyword to specify IPv4 multicast address prefixes.

	Command or Action	Purpose
		• Use the **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router-af)# neighbor SOO_group peer-group`	Creates a BGP peer group.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **soo** *extended-community-value* **Example:** `Device(config-router-af)# neighbor SOO_group soo 65000:1`	Sets the site-of-origin (SoO) value for a BGP neighbor or peer group. • In this example, the BGP peer group, SOO_group, is configured with an SoO value of 65000:1.
Step 7	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 65000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router.
Step 8	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 VRF address family with the local router.
Step 9	**neighbor** *ip-address* **peer-group** *peer-group-name* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 peer-group SOO_group`	Assigns the IP address of a BGP neighbor to a peer group.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP per Neighbor SoO Configuration

Example: Configuring a per Neighbor SoO Value Using a BGP Peer Policy Template

The following example shows how to create a peer policy template and configure an SoO value as part of the peer policy. After enabling Cisco Express Forwarding and configuring a VRF instance named SOO_VRF, a peer policy template is created and an SoO value is configured as part of the peer policy. Under the IPv4 VRF address family, a neighbor is identified and configured to inherit the peer policy that contains the SoO value.

```
ip cef
ip vrf SOO_VRF
 rd 1:1
 route-target export 1:1
 route-target import 1:1
 exit
interface GigabitEthernet 1/0/0
 ip vrf forwarding SOO_VRF
 ip address 192.168.1.2 255.255.255.0
 exit
router bgp 50000
 template peer-policy SOO_POLICY
  soo 65000:1
  exit-peer-policy
 address-family ipv4 vrf SOO_VRF
  neighbor 192.168.1.1 remote-as 65000
  neighbor 192.168.1.1 activate
  neighbor 192.168.1.1 inherit peer-policy SOO_POLICY
  end
```

Example: Configuring a per Neighbor SoO Value with a 4-Byte AS Number Using a BGP Peer Policy Template

The following example shows how to create a peer policy template and configure an SoO value using a 4-byte autonomous system number, 1.2 in asdot format, as part of the peer policy. Under the IPv4 VRF address family, a neighbor is identified and configured to inherit the peer policy that contains the SoO value.

```
router bgp 1.2
 template peer-policy SOO_POLICY
  soo 1.2:3
  exit-peer-policy
 address-family ipv4 vrf SOO_VRF
  neighbor 192.168.3.2 remote-as 1.14
  neighbor 192.168.3.2 activate
  neighbor 192.168.3.2 inherit peer-policy SOO_POLICY
  end
```

The following example shows how to create a peer policy template and configure an SoO value using a 4-byte autonomous system number, 65538 in asplain format, as part of the peer policy. Under the IPv4 VRF address family, a neighbor is identified and configured to inherit the peer policy that contains the SoO value.

```
router bgp 65538
```

```
 template peer-policy SOO_POLICY
  soo 65538:3
  exit-peer-policy
 address-family ipv4 vrf SOO_VRF
  neighbor 192.168.3.2 remote-as 65550
  neighbor 192.168.3.2 activate
  neighbor 192.168.3.2 inherit peer-policy SOO_POLICY
  end
```

Example: Configuring a per Neighbor SoO Value Using a BGP neighbor Command

The following example shows how to configure an SoO value for a BGP neighbor. After enabling Cisco Express Forwarding and configuring a VRF instance named SOO_VRF, a neighbor is identified in the IPv4 VRF address family and an SoO value is configured for the neighbor.

```
ip cef
ip vrf SOO_VRF
 rd 1:1
 route-target export 1:1
 route-target import 1:1
 exit
interface GigabitEthernet 1/0/0
 ip vrf forwarding SOO_VRF
 ip address 192.168.2.2 255.255.255.0
 exit
router bgp 50000
 address-family ipv4 vrf SOO_VRF
  neighbor 192.168.2.1 remote-as 65000
  neighbor 192.168.2.1 activate
  neighbor 192.168.2.1 soo 65000:1
  end
```

Example: Configuring a per Neighbor SoO Value Using a BGP neighbor Command and 4-Byte Autonomous System Numbers

The following example shows how to configure an SoO value for a BGP neighbor. In this example, all BGP neighbors, route targets, and SoO values use 4-byte autonomous system numbers in asplain format. After checking that CEF is enabled, a VRF instance named SOO_VRF is configured with route targets. In a BGP router session, a neighbor is configured in the IPv4 VRF address family, and an SoO value is configured for the neighbor.

```
show ip cef
ip vrf SOO_VRF
 rd 100:200
 route-target export 1.14:1
 route-target import 1.14:1
 exit
interface Ethernet 1/0
 ip vrf forwarding SOO_VRF
 ip address 192.168.2.2 255.255.255.0
 exit
router bgp 1.2
 address-family ipv4 vrf SOO_VRF
  neighbor 192.168.2.1 remote-as 1.14
  neighbor 192.168.2.1 activate
```

```
  neighbor 192.168.2.1 soo 1.14:1
  end
```

Example: Configuring a per Neighbor SoO Value Using a BGP Peer Group

The following example shows how to configure an SoO value for a BGP peer group. After enabling Cisco Express Forwarding and configuring a VRF instance named SOO_VRF, a BGP peer group is configured in the IPv4 VRF address family, an SoO value is configured for the peer group, a neighbor is identified, and the neighbor is configured as a member of the peer group.

```
ip cef
ip vrf SOO_VRF
 rd 1:1
 route-target export 1:1
 route-target import 1:1
 exit
interface GigabitEthernet 1/0/0
 ip vrf forwarding SOO_VRF
 ip address 192.168.1.2 255.255.255.0
 exit
router bgp 50000
 address-family ipv4 vrf SOO_VRF
  neighbor SOO_GROUP peer-group
  neighbor SOO_GROUP soo 65000:65
  neighbor 192.168.1.1 remote-as 65000
  neighbor 192.168.1.1 activate
  neighbor 192.168.1.1 peer-group SOO_GROUP
  end
```

Additional References

Related Documents

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

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 BGP per Neighbor SoO 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 46: Feature Information for BGP per Neighbor SoO Configuration

Feature Name	Releases	Feature Information
BGP per Neighbor SoO Configuration	12.2(33)SB 12.2(33)SRB 12.4(11)T 15.0(1)SY	The BGP per neighbor SOO configuration feature simplifies the configuration of the site-of-origin (SoO) parameter. In earlier releases, the SoO parameter is configured using an inbound route map that sets the SoO value during the update process. The per neighbor SoO configuration introduces two new commands that can be configured in submodes under router configuration mode to set the SoO value. The following commands were introduced by this feature: **neighbor soo**, and **soo**.

CHAPTER 34

BGP Next Hop Unchanged

In an external BGP (eBGP) session, by default, the router changes the next hop attribute of a BGP route (to its own address) when the router sends out a route. The BGP Next Hop Unchanged feature allows BGP to send an update to an eBGP multihop peer with the next hop attribute unchanged.

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 Next Hop Unchanged

BGP Next Hop Unchanged

In an external BGP (eBGP) session, by default, the router changes the next hop attribute of a BGP route (to its own address) when the router sends out a route. If the BGP Next Hop Unchanged feature is configured, BGP will send routes to an eBGP multihop peer without modifying the next hop attribute. The next hop attribute is unchanged.

Note There is an exception to the default behavior of the router changing the next hop attribute of a BGP route when the router sends out a route. When the next hop is in the same subnet as the peering address of the eBGP peer, the next hop is not modified. This is referred to as third party next-hop.

The BGP Next Hop Unchanged feature provides flexibility when designing and migrating networks. It can be used only between eBGP peers configured as multihop. It can be used in a variety of scenarios between two autonomous systems. One scenario is when multiple autonomous systems are connected that share the same IGP, or at least the routers have another way to reach each other's next hops (which is why the next hop can remain unchanged).

A common use of this feature is to configure Multiprotocol Label Switching (MPLS) inter-AS with multihop MP-eBGP for VPNv4 between RRs.

Another common use of this feature is a VPNv4 inter-AS Option C configuration, as defined in RFC4364, Section 10. In this configuration, VPNv4 routes are passed among autonomous systems between RR of different autonomous systems. The RRs are several hops apart, and have **neighbor next-hop unchanged** configured. PEs of different autonomous systems establish an LSP between them (via a common IGP or by advertising the next-hops--that lead to the PEs--via labeled routes among the ASBRs--routes from different autonomous systems separated by one hop). PEs are able to reach the next hops of the PEs in another AS via the LSPs, and can therefore install the VPNv4 routes in the VRF RIB.

In Cisco IOS XE Denali 16.3 release, the set ip next-hop unchanged/next-hop-unchanged allpaths IPv4/IPv6 feature extends the support for BGP Next Hop Unchanged to specific prefixes, specific next-hop, and iBGP and eBGP path prefixes while sending to eBGP neighbor. With the set ip next-hop unchanged/next-hop-unchanged allpaths IPv4/IPv6 feature you can configure unchanged next-hop for specific prefixes, for iBGP path prefixes while sending to eBGP neighbor, and can also be configured to set peer-address or a specific address as next-hop.

Restriction

The BGP Next Hop Unchanged feature can be configured only between multihop eBGP peers. The following error message will be displayed if you try to configure this feature for a directly connected neighbor:

```
%BGP: Can propagate the nexthop only to multi-hop EBGP neighbor
```

How to Configure BGP Next Hop Unchanged

Configuring the BGP Next Hop Unchanged for an eBGP Peer

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family** {**ipv4** | **ipv6** | **l2vpn** | **nsap** | **rtfilter** | **vpnv4** | **vpnv6**}
5. **neighbor** *ip-address* **remote-as** *as-number*
6. **neighbor** *ip-address* **activate**

7. **neighbor** *ip-address* **ebgp-multihop** *ttl*
8. **neighbor** *ip-address* **next-hop-unchanged**
9. **end**
10. **show ip bgp**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Router(config)# router bgp 65535`	Enters router configuration mode, and creates a BGP routing process.
Step 4	**address-family** {**ipv4** | **ipv6** | **l2vpn** | **nsap** | **rtfilter** | **vpnv4** | **vpnv6**} **Example:** `Router(config-router-af)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address family specific configurations.
Step 5	**neighbor** *ip-address* **remote-as** *as-number* **Example:** `Router(config-router-af)# neighbor 10.0.0.100 remote-as 65600`	Adds an entry to the BGP neighbor table.
Step 6	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af)# neighbor 10.0.0.100 activate`	Enables the exchange of information with the peer.
Step 7	**neighbor** *ip-address* **ebgp-multihop** *ttl* **Example:** `Router(config-router-af)# neighbor 10.0.0.100 ebgp-multihop 255`	Configures the local router to accept and initiate connections to external peers that reside on networks that are not directly connected.
Step 8	**neighbor** *ip-address* **next-hop-unchanged** **Example:**	Configures the router to send BGP updates to the specified eBGP peer without modifying the next hop attribute.

	Command or Action	Purpose
	`Router(config-router-af)# neighbor 10.0.0.100 next-hop-unchanged`	
Step 9	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode, and enters privileged EXEC mode.
Step 10	**show ip bgp** **Example:** `Router# show ip bgp`	(Optional) Displays entries in the BGP routing table. • The output will indicate if the **neighbor next-hop-unchanged** command has been configured for the selected address.

Configuration Example for BGP Next Hop Unchanged

Example: BGP Next Hop Unchanged for an eBGP Peer

The following example configures a multihop eBGP peer at 10.0.0.100 in a remote AS. When the local router sends updates to that peer, it will send them without modifying the next hop attribute.

```
router bgp 65535
 address-family ipv4
 neighbor 10.0.0.100 remote-as 65600
 neighbor 10.0.0.100 activate
 neighbor 10.0.0.100 ebgp-multihop 255
 neighbor 10.0.0.100 next-hop-unchanged
 end
```

Note All address families, such as IPv4, IPv6, VPNv4, VPNv6, L2VPN, and so on support the **next-hop unchanged** command. However, for the address family L2VPN BGP VPLS signaling, you must use the **next-hop self** command for its proper functioning.

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

Related Topic	Document Title
BGP Outbound Route Map on Route Reflector to Set IP Next Hop for iBGP Peer	"Configuring Internal BGP Features" in the *IP Routing: BGP Configuration Guide*

Technical Assistance

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

Feature Information for BGP Next Hop Unchanged

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 47: Feature Information for BGP Next Hop Unchanged

Feature Name	Releases	Feature Configuration Information
BGP Next Hop Unchanged	12.0(22)S 12.0(16)ST 12.2 12.2(14)S 15.0(1)S	The BGP Next Hop Unchanged feature allows BGP to send an update to an eBGP multihop peer with the next hop attribute unchanged. The following command was introduced by this feature: **neighbor next-hop-unchanged**.

CHAPTER 35

Detecting and Mitigating a BGP Slow Peer

The BGP Slow Peer feature allows a network administrator to detect a BGP slow peer and also to configure a peer as a slow peer statically or to dynamically mark it.

- BGP slow peer detection identifies a BGP peer that is not transmitting update messages within a configured amount of time. It is helpful to know if there is a slow peer, which indicates there is a network issue, such as network congestion or a receiver not processing updates in time, that the network administrator can address.
- BGP slow peer configuration moves or splits the peer from its normal update group to a slow update group, thus allowing the normal update group to function without being slowed down and to converge quickly.

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 Detecting and Mitigating a BGP Slow Peer

BGP Slow Peer Problem

BGP update generation uses the concept of update groups to optimize performance. An update group is a collection of peers with the identical outbound policy. When generating updates, the group policy is used to format messages that are then transmitted to the members of the group.

In order to maintain fairness in resource utilization, each update group is allocated a quota of formatted messages that it keeps in its cache. Messages are added to the cache when they are formatted by the group, and they are removed when they are transmitted to all the members of the group.

A slow peer is a peer that cannot keep up with the rate at which the Cisco IOS software is generating update messages, and is not keeping up over a prolonged period (in the order of a few minutes). There are several causes of a peer being slow:

- There is packet loss or high traffic on the link to the peer, and the throughput of the BGP TCP connection is very low.
- The peer has a heavy CPU load and cannot service the TCP connection at the required frequency.

When a slow peer is present in an update group, the number of formatted updates pending transmission builds up. When the cache limit is reached, the group does not have any more quotas to format new messages. In order for a new message to be formatted, some of the existing messages must be transmitted by the slow peer and then removed from the cache. The rest of the members of the group that are faster than the slow peer and have completed transmission of the formatted messages will not have anything new to send, even though there may be newly modified BGP networks waiting to be advertised or withdrawn. This effect of blocking formatting of all the peers in a group when one of the peers is slow in consuming updates is the "slow peer" problem.

Temporary Slowness Does Not Constitute a Slow Peer

Events that cause large churn in the BGP table (such as connection resets) can cause a brief spike in the rate of update generation. A peer that temporarily falls behind during such events, but quickly recovers after the event, is not considered a slow peer. In order for a peer for be marked as slow, it must be incapable of keeping up with the average rate of generated updates over a longer period (in the order of a few minutes).

BGP Slow Peer Feature

The BGP Slow Peer feature provides you, the network administrator, with three options:

- You can configure BGP slow peer detection only, which will simply detect a slow peer and provide you with information about it. Such detection is a key feature, especially in a large network of BGP peers, because you can then address the network problem that is causing the slow peer.
- You can configure a dynamic BGP slow peer. When such slow peer protection is configured, slow peer *detection* is enabled by default. The slow peer is moved or "split" from its normal update group to a slow update group, thus allowing the normal update group to function without being slowed down, and to converge more quickly than it would with the slow peer. You have the choice of whether to keep the slow peer in that slow update group until you clear the slow peer (by specifying the **permanent** keyword), or allow the slow peer to dynamically move back to its regular update group as conditions improve. We

recommend that you use the **permanent** keyword and resolve the network issue before you clear the slow peer status.

- You can configure a static BGP slow peer if you already know which peer is slow, perhaps due to a link issue or slow CPU process power. No detection is necessary, and it is more likely that the slow peer will remain there, hence the static configuration.

BGP Slow Peer Detection

You can choose to detect a BGP slow peer, whether or not you also configure the slow peer to be moved to a slow peer update group. Simply detecting a BGP slow peer provides you with useful information about the slow peer without splitting the update group. You should then address the network problem causing the slow peer.

Timestamp on an Update Message

BGP slow peer detection relies on the timestamp on the update messages in an update group. Update messages are timestamped when they are formatted. When BGP slow peer detection is configured, the timestamp of the oldest message in a peers queue is compared to the current time to determine if the peer is lagging more than the configured slow peer time threshold.

For example, if the oldest message in the peers queue was formatted more than 3 minutes ago, but the BGP slow peer detection threshold is configured at 3 minutes, then the peer that formatted that update message is determined to be a slow peer.

The Cisco IOS software generates a syslog event when a slow peer is detected or recovered (when its update group has converged and it has no messages formatted before the threshold time).

Benefit of BGP Slow Peer Detection

Slow peer detection provides you with information about the slow peer, and you can resolve the root cause without moving the peer to a different update group. Therefore, slow peer detection requires just one command that helps you identify something in your network that could be improved.

Benefits of Configuring a Dynamic or Static BGP Slow Peer

When a slow peer is present in an update group, the number of formatted updates pending transmission builds up. New messages cannot be formatted and transmitted until the backlog is reduced. That scenario delays BGP update packets and therefore delays BGP networks from being advertised. The problem can be resolved or prevented by configuring a dynamic slow peer or a static slow peer. Such configuration causes a slow peer to be put into a new, slow peer update group and thus prevents the slow peer from delaying the BGP peers that are not slow.

Static Slow Peer

If you believe that a peer is slow, you can statically configure the peer to be a slow peer. A static slow peer is recommended for a peer that is known to be slow, perhaps having a slow link or low processing power.

Static slow peer configuration causes the Cisco IOS software to create a separate update group for the peer. If you configure two peers belonging to the same update group as slow, these two peers will be moved into a single slow peer update group because their policy will match. The slow update group will function at the pace of the slowest of the slow peers.

A static slow peer can be configured in either of two ways:

- At the BGP neighbor (address family) level
- Via a peer policy template

You probably want to determine the root cause of the peer being slow, such as network congestion or a receiver not processing updates in time. A static slow peer is not automatically restored to its original update group. You can restore a static slow peer to its original update group by using the **no neighbor slow-peer split-update-group static**command or the **no slow-peer split-update-group static**command.

Dynamic Slow Peer

An alternative to marking a static slow peer is to configure slow peers dynamically, based on the amount of time that the timestamp of the oldest message in a peers queue lags behind the current time. The default threshold is 300 seconds, and is configurable. We recommend that you specify the optional **permanent** keyword, which causes the peer to remain in the slow peer group while you resolve the root cause of the slow peer. You can then use the **clear bgp slow** command to move the peer back to its original group.

If you do not configure the **permanent** keyword, the peer moves back to its original group if and when it regains its non-slow functioning.

When a dynamic slow peer is configured, detection is enabled automatically.

You can configure dynamic slow peers in three ways:

- At the address family view level
- At the neighbor topology (that is, neighbor address-family) level
- Via a peer policy template

How to Detect and Mitigate a BGP Slow Peer

Detecting a Slow Peer

You might want to just detect a slow peer, but not move the slow peer out of its update group. Such detection notifies you by way of a syslog message that a BGP peer is not transmitting update messages within a configurable amount of time. The peer remains in its update group; the update group is not split. The syslog message level is notice level for both detection and recovery.

If you want to dynamically configure a BGP slow peer, see the Configuring Dynamic Slow Peer Protection, on page 616. You will notice that that task includes and requires the step of detecting a slow peer.

Detect a slow peer by performing one of the following tasks:

Detecting Dynamic Slow Peers at the Address-Family Level

Perform this task to detect all dynamic slow peers at the address-family level. (If you want to detect *specific* slow peers, detect slow peers at the neighbor level or by using a peer policy template).

The last step is optional; use it if you want to disable slow peer detection for a specific peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number*
5. **address-family ipv4**
6. **bgp slow-peer detection** [**threshold** *seconds*]
7. **neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer detection disable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 10.4.4.4 remote-as 5`	(Optional) Adds an entry to the BGP or multiprotocol BGP neighbor table. • This step is required if you intend to disable dynamic slow peer protection for a specific peer as shown in Step 7 below.
Step 5	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 6	**bgp slow-peer detection** [**threshold** *seconds*] **Example:** `Router(config-router-af)# bgp slow-peer detection threshold 600`	Configures global slow peer detection and specifies the time in seconds that the timestamp of the oldest update message in a peers queue can be lagging behind the current time before the peer is determined to be a slow peer. • The range of the threshold is from 120 to 3600. As long as the command is configured, the default is 300.

	Command or Action	Purpose
Step 7	**neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer detection disable** **Example:** `Router(config-router-af)# neighbor 10.4.4.4 slow-peer detection disable`	(Optional) Disables slow-peer detection for a specific peer. • Use this command only if you have configured global slow peer detection in Step 5, and now you want to disable slow peer detection for a specific peer or peer group.

Detecting Dynamic Slow Peers at the Neighbor Level

Perform this task to detect dynamic slow peers at a specific neighbor address or belonging to a specific peer group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4**
5. **neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer detection**[**threshold** *seconds*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 5	**neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer detection**[**threshold** *seconds*] **Example:**	(Optional) Specifies the time in seconds that the timestamp of the oldest message in a peers queue can be lagging behind the current time before the peer is determined to be a slow peer.

	Command or Action	Purpose
	`Router(config-router-af)# neighbor 172.60.2.3 slow-peer detection threshold 1200`	• The range of the threshold is 120 seconds to 3600 seconds. As long as the command is configured, the default is 300 seconds.

Detecting Dynamic Slow Peers Using a Peer Policy Template

Perform the following task to detect BGP slow peers using a peer policy template.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-policy** *policy-template-name*
5. **slow-peer detection** [**threshold** *seconds*]
6. **exit**
7. **address-family ipv4**
8. **neighbor** *ip-address* **inherit peer-policy** *policy-template-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**template peer-policy** *policy-template-name* **Example:** `Router(config-router)# template peer-policy global`	Enters policy template configuration mode and creates a peer policy template.

	Command or Action	Purpose
Step 5	**slow-peer detection** [**threshold** *seconds*] **Example:** `Router(config-router-ptmp)# slow-peer detection threshold 600`	Specifies the time in seconds that the timestamp of the oldest message in a peers queue can be lagging behind the current time before the peer is determined to be a slow peer. • The range of the threshold is from 120 to 3600. As long as the command is configured, the default is 300.
Step 6	**exit** **Example:** `Router(config-router-ptmp)# exit`	Exits to higher configuration mode.
Step 7	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 8	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Router(config-router-af)# neighbor 10.0.0.1 inherit peer-policy global`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration.

Marking a Peer as a Static Slow Peer

There are two ways to statically configure a slow peer. Perform one of the following tasks in this section to statically configure a slow peer:

Marking a Peer as a Static Slow Peer at the Neighbor Level

Perform this task to configure a static slow peer at a specific neighbor address or belonging to a specific peer group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4**
5. **neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group static**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 5	**neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group static** **Example:** `Router(config-router-af)# neighbor 172.16.1.1 slow-peer split-update-group static`	Configures the neighbor at the specified address as a slow peer. • Use the **no neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group static** command if you want to restore the peer to its original, non-slow update group.

Marking a Peer as a Static Slow Peer Using a Peer Policy Template

Perform this task to configure a static slow peer by using a peer policy template.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-policy** *policy-template-name*
5. **slow-peer split-update-group static**
6. **exit**
7. **address-family ipv4**
8. **neighbor** *ip-address* **inherit peer-policy** *policy-template-name*

DETAILED STEPS

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

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**template peer-policy** *policy-template-name* **Example:** `Router(config-router)# template peer-policy global`	Enters policy template configuration mode and creates a peer policy template.
Step 5	**slow-peer split-update-group static** **Example:** `Router(config-router-ptmp)# slow-peer split-update-group static`	Configures the neighbor at the specified address as a slow peer. • Use the **no slow-peer split-update-group static** command if you want to restore the peer to its normal status.
Step 6	**exit** **Example:** `Router(config-router-ptmp)# exit`	Exits to higher configuration mode.
Step 7	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 8	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Router(config-router-af)# neighbor 10.0.0.1 inherit peer-policy global`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration.

Configuring Dynamic Slow Peer Protection

There are three ways to dynamically configure slow peers, also known as slow peer protection. Perform one or more of the tasks in this section to configure dynamic slow peers:

Configuring Dynamic Slow Peers at the Address-Family Level

Configuring dynamic slow peers at the address-family level applies to all peers in the address family specified. (If you want to configure *specific* slow peers, perform this task at the neighbor level or by using a peer policy template.)

The last step is optional; perform it only if you want to disable slow peer protection for a specific peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number*
5. **address-family ipv4**
6. **bgp slow-peer detection** [**threshold** *seconds*]
7. **bgp slow-peer split-update-group dynamic** [**permanent**]
8. **neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group dynamic disable**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**neighbor** {*ip-address* | *ipv6-address*[**%**] | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 10.4.4.4 remote-as 5`	(Optional) Adds an entry to the BGP or multiprotocol BGP neighbor table. • This step is required if you intend to disable dynamic slow peer protection for a specific peer as shown in Step 8 below.
Step 5	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.

	Command or Action	Purpose
Step 6	**bgp slow-peer detection** [**threshold** *seconds*] **Example:** `Router(config-router-af)# bgp slow-peer detection threshold 600`	(Optional) Specifies the time in seconds that the timestamp of the oldest update message in a peers queue can be lagging behind the current time before the peer is determined to be a slow peer. • When a dynamic slow peer is configured, as in the next step, this detection is enabled automatically. • The range of the threshold is from 120 to 3600. The default is 300.
Step 7	**bgp slow-peer split-update-group dynamic** [**permanent**] **Example:** `Router(config-router-af)# bgp slow-peer split-update-group dynamic permanent`	Moves the dynamically detected slow peer to a slow update group. • If a static slow peer update group exists (because of a static slow peer), the dynamic slow peer will be moved to the static slow peer update group. • If no static slow peer update group exists, a new slow peer update group will be created and the peer will be moved to that. • We recommend using the **permanent** keyword. If the **permanent** keyword is used, the peer will not be moved to its original update group automatically. After you determine the root cause of the slowness, such as network congestion, for example, you can use a **clear bgp slow**command to move the peer to its original update group. See the Restoring Dynamic Slow Peers as Normal Peers, on page 622 to move a dynamically slow peer back to its original update group. • If the **permanent** keyword is not used, the slow peer will be moved back to its regular original update group after it becomes a normal peer (converges).
Step 8	**neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group dynamic disable** **Example:** `Router(config-router-af)# neighbor 10.4.4.4 slow-peer split-update-group dynamic disable`	(Optional) Perform this step only if you want to disable dynamic slow peer protection for a specific peer.

Configuring Dynamic Slow Peers at the Neighbor Level

Perform this task to configure a dynamic slow peer at a specific neighbor address or belonging to a specific peer group.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4**
5. **neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer detection**[**threshold** *seconds*]
6. **neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group dynamic** [**permanent**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.
Step 4	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.
Step 5	**neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer detection**[**threshold** *seconds*] **Example:** `Router(config-router-af)# neighbor 172.60.2.3 slow-peer detection threshold 1200`	(Optional) Specifies the time in seconds that the timestamp of the oldest update message in a peers queue can be lagging behind the current time before the peer is determined to be a slow peer. • When a dynamic slow peer is configured, as in the next step, this detection is enabled automatically. • The range of the threshold is from 120 to 3600. The default is 300.
Step 6	**neighbor** {*neighbor-address* | *peer-group-name*} **slow-peer split-update-group dynamic** [**permanent**] **Example:** `Router(config-router-af)# neighbor 172.60.2.3 slow-peer split-update-group dynamic permanent`	Moves the dynamically detected slow peer to a slow update group. • If a static slow peer update group exists (because of a static slow peer), the dynamic slow peer will be moved to the static slow peer update group. • If no static slow peer update group exists, a new slow peer update group will be created and the peer will be moved to that.

	Command or Action	Purpose
		• We recommend using the **permanent** keyword. If the **permanent** keyword is used, the peer will not be moved to its original update group automatically. After you resolve the root cause of the slowness, such as network congestion, for example, you can use a **clear bgp slow** command to move the peer to its original update group. See the Restoring Dynamic Slow Peers as Normal Peers, on page 622 to move a dynamically slow peer back to its original update group. • If the **permanent** keyword is not used, the slow peer will be moved back to its regular original update group after it becomes a normal peer (converges).

Configuring Dynamic Slow Peers Using a Peer Policy Template

Perform this task to configure a BGP slow peer using a peer policy template.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **template peer-policy** *policy-template-name*
5. **slow-peer detection** [**threshold** *seconds*]
6. **slow-peer split-update-group dynamic** [**permanent**]
7. **exit**
8. **address-family ipv4**
9. **neighbor** *ip-address* **inherit peer-policy** *policy-template-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 5`	Configures the BGP routing process.

	Command or Action	Purpose
Step 4	**template peer-policy** *policy-template-name* **Example:** `Router(config-router)# template peer-policy global`	Enters policy template configuration mode and creates a peer policy template.
Step 5	**slow-peer detection** [**threshold** *seconds*] **Example:** `Router(config-router-ptmp)# slow-peer detection threshold 600`	(Optional) Specifies the time in seconds that the timestamp of the oldest message in a peers queue can be lagging behind the current time before the peer is determined to be a slow peer. • When a dynamic slow peer is configured, as in the next step, this detection is enabled automatically. • The range of the threshold is from 120 to 3600. The default is 300.
Step 6	**slow-peer split-update-group dynamic** [**permanent**] **Example:** `Router(config-router-ptmp)# slow-peer split-update-group dynamic permanent`	Moves the dynamically detected slow peer to a slow update group. • If a static slow peer update group exists (because of a static slow peer), the dynamic slow peer will be moved to the static slow peer update group. • If no static slow peer update group exists, a new slow peer update group will be created and the peer will be moved to that. • We recommend using the **permanent** keyword. If the **permanent** keyword is used, the peer will not be moved to its original update group automatically. After you determine the root cause of the slowness, such as network congestion, for example, you can use a command to move the peer to its original update group. See the Restoring Dynamic Slow Peers as Normal Peers, on page 622 to move a dynamically slow peer back to its original update group. • If the **permanent** keyword is not used, the slow peer will be moved back to its regular original update group after it becomes a normal peer (converges).
Step 7	**exit** **Example:** `Router(config-router-ptmp)# exit`	Exits to higher configuration mode.
Step 8	**address-family ipv4** **Example:** `Router(config-router)# address-family ipv4`	Enters address family configuration mode.

	Command or Action	Purpose
Step 9	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Router(config-router-af)# neighbor 10.0.0.1 inherit peer-policy global`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration.

Displaying Output About Dynamic Slow Peers

Use one or more of the **show** commands in this task to display output about dynamically configured BGP slow peers.

SUMMARY STEPS

1. **enable**
2. **show ip bgp** [**ipv4** {**multicast** | **unicast**} | **vpnv4 all** | **vpnv6 unicast all** | **topology**{*****| *routing-topology-instance-name*}] [**update-group**] **summary slow**
3. **show ip bgp** [**ipv4** {**multicast** | **unicast**} | **vpnv4 all** | **vpnv6 unicast all**] **neighbors slow**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp** [**ipv4** {**multicast** | **unicast**} | **vpnv4 all** | **vpnv6 unicast all** | **topology**{*****| *routing-topology-instance-name*}] [**update-group**] **summary slow** **Example:** `Router# show ip bgp summary slow`	Displays information about dynamic BGP slow peers in summary form.
Step 3	**show ip bgp** [**ipv4** {**multicast** | **unicast**} | **vpnv4 all** | **vpnv6 unicast all**] **neighbors slow** **Example:** `Router# show ip bgp neighbors slow`	Displays information about dynamic BGP slow peer neighbors.

Restoring Dynamic Slow Peers as Normal Peers

Once you, the network administrator, resolve the root cause of a slow peer (network congestion, or a receiver not processing updates in time, and so forth), use the **clear**commands in the following task to move the peer back to its original group. Both commands perform the same function.

Note Note that *statically* configured slow peers are not affected by these **clear** commands. To restore a statically configured slow peer to its original update group, use the **no** form of the command shown in one of the tasks in the Marking a Peer as a Static Slow Peer, on page 614.

SUMMARY STEPS

1. **enable**
2. **clear ip bgp** {[*af*] *****| *neighbor-address* | **peer-group** *group-name*} **slow**
3. **clear bgp** *af* {*****| *neighbor-address* | **peer-group** *group-name*} **slow**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip bgp** {[*af*] *****| *neighbor-address*	**peer-group** *group-name*} **slow** **Example:** `Router# clear ip bgp * slow`	(Optional) Restores neighbor(s) from a slow update peer group to their original update peer group. • *af* is one of the following address families: **ipv4**, **vpnv4**, or **vpnv6**. Moves all peers in the IPv4, VPNv4 or VPNv6 address family back to their original update groups. • ***** moves all peers back to their original update groups.
Step 3	**clear bgp** *af* {*****| *neighbor-address*	**peer-group** *group-name*} **slow** **Example:** `Router# clear bgp ipv4 * slow`	(Optional) Restores neighbor(s) from slow update peer group to their original update peer group. • *af* is one of the following address families: **ipv4**, **vpnv4**, or **vpnv6**. Moves peers in the IPv4, VPNv4 or VPNv6 address family back to their original update groups. • ***** moves all peers in the address family back to their original update groups.

Configuration Examples for Detecting and Mitigating a BGP Slow Peer

Example: Static Slow Peer

The following example marks the neighbor at 192.168.12.10 as a static slow peer.

```
router bgp 5
address-family ipv4
 neighbor 192.168.12.10 slow-peer split-update-group static
```

Example: Static Slow Peer Using Peer Policy Template

The following example configures a static slow peer using a peer policy template named ipv4_ucast_pp2. The neighbor at 10.0.101.4 inherits the policy.

```
router bgp 13
template peer-policy ipv4_ucast_pp2
slow-peer split-update-group static
exit-peer-policy
!
no bgp default route-target filter
no bgp enforce-first-as
bgp log-neighbor-changes
neighbor 10.0.101.4 remote-as 13
address-family ipv4
neighbor 10.0.101.4 inherit peer-policy ipv4_ucast_pp2

RouterA# show ip bgp template peer-policy ipv4_ucast_pp2

Template:ipv4_ucast_pp2, index:2.
Local policies:0x180000000, Inherited polices:0x0
Local disable policies:0x0, Inherited disable policies:0x0
Locally configured policies:
  slow-peer split-update-group static
  Inherited policies:
```

Example: Dynamic Slow Peer at the Neighbor Level

The following example configures a slow peer at the neighbor level. The neighbor at 10.0.101.3 is configured with dynamic slow peer protection at a default threshold of 300 seconds.

```
router bgp 13
no bgp default route-target filter
no bgp enforce-first-as
bgp log-neighbor-changes
 neighbor 10.0.101.3 remote-as 13
address-family ipv4
 neighbor 10.0.101.3 slow-peer split-update-group dynamic
```

Example: Dynamic Slow Peers Using Peer Policy Template

In the following example, Router A uses a peer policy template named ipv4_ucast_pp1 and sets a detection threshold of 120 seconds. The **permanent** keyword causes slow peers to remain in the slow update group until the network administrator uses the **clear ip bgp slow** command to move the peer to its original update group. The neighbor at 10.0.101.2 inherits the peer policy, which means that if that neighbor is determined to be slow, it is moved to a slow update group.

```
router bgp 13
 template peer-policy ipv4_ucast_pp1
 slow-peer detection threshold 120
 slow-peer split-update-group dynamic permanent
 exit-peer-policy
!
no bgp default route-target filter
no bgp enforce-first-as
bgp log-neighbor-changes
 neighbor 10.0.101.2 remote-as 13
!
address-family ipv4
 neighbor 10.0.101.2 activate
 neighbor 10.0.101.2 inherit peer-policy ipv4_ucast_pp1
```

The following output displays the locally configured policies.

```
RouterA# show ip bgp template peer-policy ipv4_ucast_pp1

Template:ipv4_ucast_pp1, index:1.
Local policies:0x300000000, Inherited polices:0x0
Local disable policies:0x0, Inherited disable policies:0x0
Locally configured policies:
  slow-peer detection threshold is 120
  slow-peer split-update-group dynamic permanent
Inherited policies:
```

Example: Dynamic Slow Peers Using Peer Group

The following example configures two peer groups: ipv4_ucast_pg1 and ipv4_ucast_pg2. The neighbor at 10.0.101.1 belongs to ipv4_ucast_pg1, where slow peer detection is configured for 120 seconds. The neighbor at 10.0.101.5 belongs to ipv4_ucast_pg2, where slow peer detection is configured at 140 seconds.

```
router bgp 13
no bgp default route-target filter
no bgp enforce-first-as
bgp log-neighbor-changes
 neighbor ipv4_ucast_pg1 peer-group
 neighbor ipv4_ucast_pg2 peer-group
 neighbor ipv4_ucast_pg1 remote-as 13
 neighbor ipv4_ucast_pg2 remote-as 13
 neighbor 10.0.101.1 peer-group ipv4_ucast_pg1
 neighbor 10.0.101.5 peer-group ipv4_ucast_pg2
address-family ipv4
 neighbor ipv4_ucast_pg1 slow-peer detection threshold 120
 neighbor ipv4_ucast_pg1 slow-peer split-update-group dynamic
 neighbor ipv4_ucast_pg2 slow-peer detection threshold 140
 neighbor ipv4_ucast_pg2 slow-peer split-update-group dynamic
```

The following output displays information about the peer group ipv4_ucast_pg1.

```
RouterA# show ip bgp peer-group ipv4_ucast_pg1

BGP peer-group is ipv4_ucast_pg1,  remote AS 13
  BGP version 4
  Neighbor sessions:
    0 active, is multisession capable
  Default minimum time between advertisement runs is 0 seconds
 For address family: IPv4 Unicast
  BGP neighbor is ipv4_ucast_pg1, peer-group internal, members:
  10.0.101.1
  Index 0
  Slow-peer detection is enabled, threshold value is 120
  Slow-peer split-update-group dynamic is enabled
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

The following output displays information about the peer group ipv4_ucast_pg2.

```
RouterA# show ip bgp peer-group ipv4_ucast_pg2

BGP peer-group is ipv4_ucast_pg2,  remote AS 13
  BGP version 4
  Neighbor sessions:
    0 active, is multisession capable
  Default minimum time between advertisement runs is 0 seconds
 For address family: IPv4 Unicast
  BGP neighbor is ipv4_ucast_pg2, peer-group internal, members:
  10.0.101.5
  Index 0
  Slow-peer detection is enabled, threshold value is 140
  Slow-peer split-update-group dynamic is enabled
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

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
Syslog messages and the **logging console** command	*Cisco IOS Network Management Command Reference*
BGP peer policy templates	Configure a Basic BGP Network module in the *Cisco IOS IP Routing: BGP Configuration Guide*

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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 Detecting and Mitigating a BGP Slow Peer

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 48: Feature Information for Detecting and Mitigating a BGP Slow Peer

Feature Name	Releases	Feature Information
BGP Slow Peer	Cisco IOS 15.0(1)MR	The BGP Slow Peer feature allows a network administrator to detect a BGP slow peer and also to configure a peer as a slow peer statically or dynamically mark it. • BGP slow peer detection identifies a BGP peer that is not transmitting update messages within a configured amount of time. It is helpful to know if there is a slow peer, which indicates there is a network issue that the network administrator can address. • BGP slow peer configuration causes the peer to be moved from its normal update group to a slow update group, thus allowing the normal update group to function without being slowed down and to converge quickly. The following commands were modified: • **clear ip bgp** • **show ip bgp neighbors** • **show ip bgp summary** The following commands were introduced: • **bgp slow-peer detection** • **bgp slow-peer split-update-group dynamic** • **neighbor slow-peer detection** • **neighbor slow-peer split-update-group dynamic** • **neighbor slow-peer split-update-group static** • **slow-peer detection** • **slow-peer split-update-group dynamic** • **slow-peer split-update-group static**

CHAPTER 36

Per-VRF Assignment of BGP Router ID

The Per-VRF Assignment of BGP Router ID feature introduces the ability to have VRF-to-VRF peering in Border Gateway Protocol (BGP) on the same router. BGP is designed to refuse a session with itself because of the router ID check. The per-VRF assignment feature allows a separate router ID per VRF using a new keyword in the existing **bgp router-id** command. The router ID can be manually configured for each VRF or can be assigned automatically either globally under address family configuration mode or for each VRF.

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 Per-VRF Assignment of BGP Router ID

Before you configure this feature, Cisco Express Forwarding (CEF) or distributed CEF (dCEF) must be enabled in the network, and basic BGP peering is assumed to be running in the network.

Information About Per-VRF Assignment of BGP Router ID

BGP Router ID

The BGP router identifier (ID) is a 4-byte field that is set to the highest IP address on the router. Loopback interface addresses are considered before physical interface addresses because loopback interfaces are more stable than physical interfaces. The BGP router ID is used in the BGP algorithm for determining the best path to a destination where the preference is for the BGP router with the lowest router ID. It is possible to manually configure the BGP router ID using the **bgp router-id** command to influence the best path algorithm.

Per-VRF Router ID Assignment

The Per-VRF Assignment of BGP Router ID feature introduces the ability to have VRF-to-VRF peering in Border Gateway Protocol (BGP) on the same router. BGP is designed to refuse a session with itself because of the router ID check. The Per-VRF Assignment of BGP Router ID feature allows a separate router ID per VRF using a new keyword in the existing **bgp router-id** command. The router ID can be manually configured for each VRF or can be assigned automatically either globally under address family configuration mode or for each VRF.

Route Distinguisher

A route distinguisher (RD) creates routing and forwarding tables and specifies the default route distinguisher for a VPN. The RD is added to the beginning of an IPv4 prefix to change it into a globally unique VPN-IPv4 prefix. An RD can be composed in one of two ways: with an autonomous system number and an arbitrary number or with an IP address and an arbitrary number.

You can enter an RD in either of these formats:

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

45000:3

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

192.168.10.15:1

How to Configure Per-VRF Assignment of BGP Router ID

Configuring VRF Instances

Perform this task to configure VRF instances to be used with the Per-VRF Assignment of Router ID tasks. In this task, a VRF instance named vrf_trans is created. To make the VRF functional, a route distinguisher is created. When the route distinguisher is created, the routing and forwarding tables are created for the VRF instance named vrf_trans.

Before you begin

This task assumes that you have CEF or dCEF enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*
5. **route-target** [**import** | **both**] *route-target-ext-community*
6. **route-target** [**export** | **both**] *route-target-ext-community*
7. **exit**
8. Repeat Step 3 through Step 7 for each VRF to be defined.

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf vrf_trans`	Defines a VRF instance and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 45000:2`	Creates routing and forwarding tables for a VRF and specifies the default RD for a VPN. • Use the *route-distinguisher* argument to specify the default RD for a VPN. There are two formats you can use to specify an RD. For more details, see the “Route Distinguisher” section. • In this example, the RD uses an autonomous system number with the number 2 after the colon.
Step 5	**route-target** [**import** | **both**] *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 55000:5`	Creates a route-target extended community for a VRF. • Use the **import** keyword to import routing information from the target VPN extended community.

	Command or Action	Purpose
		• Use the **both** keyword to both import routing information from and export routing information to the target VPN extended community. • Use the *route-target-ext-community* argument to specify the VPN extended community.
Step 6	**route-target** [**export** | **both**] *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 55000:1`	Creates a route-target extended community for a VRF. • Use the **export** keyword to export routing information to the target VPN extended community. • Use the **both** keyword to both import routing information from and export routing information to the target VPN extended community. • Use the *route-target-ext-community* argument to specify the VPN extended community.
Step 7	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and returns to global configuration mode.
Step 8	Repeat Step 3 through Step 7 for each VRF to be defined.	

Associating VRF Instances with Interfaces

Perform this task to associate VRF instances with interfaces to be used with the per-VRF assignment tasks. In this task, a VRF instance named vrf_trans is associated with a serial interface.

Make a note of the IP addresses for any interface to which you want to associate a VRF instance because the **ip vrf forwarding** command removes the IP address. Step 8 allows you to reconfigure the IP address.

Before you begin

- This task assumes that you have CEF or dCEF enabled.
- This task assumes that VRF instances have been configured as shown in preceding “Configuring VRF Instances” task in this module.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask* [**secondary**]
5. **exit**
6. **interface** *type number*
7. **ip vrf forwarding** *vrf-name* [**downstream** *vrf-name2*]

8. **ip address** *ip-address mask* [**secondary**]
9. Repeat Step 5 through Step 8 for each VRF to be associated with an interface.
10. **end**
11. **show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface loopback0`	Configures an interface type and enters interface configuration mode. • In this example, loopback interface 0 is configured.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 172.16.1.1 255.255.255.255`	Configures an IP address. • In this example, the loopback interface is configured with an IP address of 172.16.1.1.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**interface** *type number* **Example:** `Router(config)# interface serial2/0`	Configures an interface type and enters interface configuration mode. • In this example, serial interface 2/0 is configured.
Step 7	**ip vrf forwarding** *vrf-name* [**downstream** *vrf-name2*] **Example:** `Router(config-if)# ip vrf forwarding vrf_trans`	Associates a VRF with an interface or subinterface. • In this example, the VRF named vrf_trans is associated with serial interface 2/0. **Note** Executing this command on an interface removes the IP address. The IP address should be reconfigured.
Step 8	**ip address** *ip-address mask* [**secondary**]	Configures an IP address.

	Command or Action	Purpose
	Example: `Router(config-if)# ip address 192.168.4.1 255.255.255.0`	• In this example, serial interface 2/0 is configured with an IP address of 192.168.4.1.
Step 9	Repeat Step 5 through Step 8 for each VRF to be associated with an interface.	--
Step 10	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 11	**show ip vrf** [**brief** | **detail** | **interfaces** | **id**] [*vrf-name*] **Example:** `Router# show ip vrf interfaces`	(Optional) Displays the set of defined VRFs and associated interfaces. • In this example, the output from this command shows the VRFs that have been created and their associated interfaces.

Examples

The following output show s that two VRF instances named vrf_trans and vrf_users were configured on two serial interfaces.

```
Router# show ip vrf interfaces

Interface          IP-Address      VRF                        Protocol
Serial2            192.168.4.1     vrf_trans                  up
Serial3            192.168.5.1     vrf_user                   up
```

Manually Configuring a BGP Router ID per VRF

Perform this task to manually configure a BGP router ID for each VRF. In this task, several address family configurations are shown and the router ID is configured in the IPv4 address family mode for one VRF instance. Step 22 shows you how to repeat certain steps to permit the configuration of more than one VRF on the same router.

Before you begin

This task assumes that you have previously created the VRF instances and associated them with interfaces. For more details, see the “Configuring VRF Instances” task and the “Associating VRF Instances with Interfaces” task earlier in this module.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*

4. **no bgp default ipv4-unicast**
5. **bgp log-neighbor-changes**
6. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number*
8. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
9. **neighbor** {*ip-address* | *peer-group-name*} **activate**
10. **neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**}
11. **exit-address-family**
12. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
13. **redistribute connected**
14. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
15. **neighbor** *ip-address* **local-as** *autonomous-system-number* [**no-prepend** [**replace-as** [**dual-as**]]]
16. **neighbor** {*ip-address* | *peer-group-name*} **ebgp-multihop** [*ttl*]
17. **neighbor** {*ip-address* | *peer-group-name*} **activate**
18. **neighbor** *ip-address* **allowas-in** [*number*]
19. **no auto-summary**
20. **no synchronization**
21. **bgp router-id** {*ip-address* | **auto-assign**}
22. Repeat Step 11 to Step 21 to configure another VRF instance.
23. **end**
24. **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**no bgp default ipv4-unicast** **Example:**	Disables the IPv4 unicast address family for the BGP routing process.

	Command or Action	Purpose
	`Router(config-router)# no bgp default ipv4-unicast`	**Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.
Step 5	**bgp log-neighbor-changes** **Example:** `Router(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor is an internal neighbor.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number* **Example:** `Router(config-router)# neighbor 192.168.1.1 update-source loopback0`	Allows BGP sessions to use any operational interface for TCP connections. • In this example, BGP TCP connections for the specified neighbor are sourced with the IP address of the loopback interface rather than the best local address.
Step 8	**address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]} **Example:** `Router(config-router)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations. • The example creates a VPNv4 address family session.
Step 9	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Router(config-router-af)# neighbor 172.16.1.1 activate`	Activates the neighbor under the VPNv4 address family. • In this example, the neighbor 172.16.1.1 is activated.

	Command or Action	Purpose
Step 10	**neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**} **Example:** `Router(config-router-af)# neighbor 172.16.1.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 172.16.1.1.
Step 11	**exit-address-family** **Example:** `Router(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 12	**address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]} **Example:** `Router(config-router)# address-family ipv4 vrf vrf_trans`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations. • The example specifies that the VRF instance named vrf_trans is to be associated with subsequent IPv4 address family configuration commands.
Step 13	**redistribute connected** **Example:** `Router(config-router-af)# redistribute connected`	Redistributes from one routing domain into another routing domain. • In this example, the **connected** keyword is used to represent routes that are established automatically when IP is enabled on an interface. • Only the syntax applicable to this step is displayed. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.
Step 14	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.1.1 remote-as 40000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 192.168.1.1 is an external neighbor.
Step 15	**neighbor** *ip-address* **local-as** *autonomous-system-number* [**no-prepend** [**replace-as** [**dual-as**]]]	Customizes the AS_PATH attribute for routes received from an eBGP neighbor.

	Command or Action	Purpose
	Example: `Router(config-router-af)# neighbor 192.168.1.1 local-as 50000 no-prepend`	• The autonomous system number from the local BGP routing process is prepended to all external routes by default. • Use the **no-prepend** keyword to not prepend the local autonomous system number to any routes received from the eBGP neighbor. • In this example, routes from the neighbor at 192.168.1.1 will not contain the local autonomous system number.
Step 16	**neighbor** {*ip-address* | *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Router(config-router-af)# neighbor 192.168.1.1 ebgp-multihop 2`	Accepts and attempts BGP connections to external peers residing on networks that are not directly connected. • In this example, BGP is configured to allow connections to or from neighbor 192.168.1.1, which resides on a network that is not directly connected.
Step 17	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Router(config-router-af)# neighbor 192.168.1.1 activate`	Activates the neighbor under the IPV4 address family. • In this example, the neighbor 192.168.1.1 is activated.
Step 18	**neighbor** *ip-address* **allowas-in** [*number*] **Example:** `Router(config-router-af)# neighbor 192.168.1.1 allowas-in 1`	Configures provider edge (PE) routers to allow the readvertisement of all prefixes that contain duplicate autonomous system numbers. • In the example, the PE router with autonomous system number 45000 is configured to allow prefixes from the VRF vrf-trans. The neighboring PE router with the IP address 192.168.1.1 is set to be readvertised once to other PE routers with the same autonomous system number.
Step 19	**no auto-summary** **Example:** `Router(config-router-af)# no auto-summary`	Disables automatic summarization and sends subprefix routing information across classful network boundaries.
Step 20	**no synchronization** **Example:** `Router(config-router-af)# no synchronization`	Enables the Cisco IOS software to advertise a network route without waiting for synchronization with an Internal Gateway Protocol (IGP).
Step 21	**bgp router-id** {*ip-address* | **auto-assign**} **Example:**	Configures a fixed router ID for the local BGP routing process.

	Command or Action	Purpose
	`Router(config-router-af)# bgp router-id 10.99.1.1`	• In this example, the specified BGP router ID is assigned for the VRF instance associated with this IPv4 address family configuration.
Step 22	Repeat Step 11 to Step 21 to configure another VRF instance.	--
Step 23	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 24	**show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} **Example:** `Router# show ip bgp vpnv4 all`	(Optional) Displays VPN address information from the BGP table. • In this example, the complete VPNv4 database is displayed. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS Multiprotocol Label Switching Command Reference*.

Examples

The following sample output assumes that two VRF instances named vrf_trans and vrf_user were configured each with a separate router ID. The router ID is shown next to the VRF name.

```
Router# show ip bgp vpnv4 all

BGP table version is 5, 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
Route Distinguisher: 1:1 (default for vrf vrf_trans) VRF Router ID 10.99.1.2
*> 192.168.4.0      0.0.0.0                  0         32768 ?
Route Distinguisher: 42:1 (default for vrf vrf_user) VRF Router ID 10.99.1.1
*> 192.168.5.0      0.0.0.0                  0         32768 ?
```

Automatically Assigning a BGP Router ID per VRF

Perform this task to automatically assign a BGP router ID for each VRF. In this task, a loopback interface is associated with a VRF and the **bgp router-id** command is configured at the router configuration level to automatically assign a BGP router ID to all VRF instances. Step 9 shows you how to repeat certain steps to configure each VRF that is to be associated with an interface. Step 30 shows you how to configure more than one VRF on the same router.

Before you begin

This task assumes that you have previously created the VRF instances as shown in the "Configuring VRF Instances" task in this module.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *ip-address mask* [**secondary**]
5. **exit**
6. **interface** *type number*
7. **ip vrf forwarding** *vrf-name* [**downstream** *vrf-name2*]
8. **ip address** *ip-address mask* [**secondary**]
9. Repeat Step 5 through Step 8 for each VRF to be associated with an interface.
10. **exit**
11. **router bgp** *autonomous-system-number*
12. **bgp router-id** {*ip-address* | **vrf auto-assign**}
13. **no bgp default ipv4-unicast**
14. **bgp log-neighbor-changes**
15. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
16. **neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number*
17. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
18. **neighbor** {*ip-address* | *peer-group-name*} **activate**
19. **neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**}
20. **exit-address-family**
21. **address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]}
22. **redistribute connected**
23. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number*
24. **neighbor** *ip-address* **local-as** *autonomous-system-number* [**no-prepend** [**replace-as** [**dual-as**]]]
25. **neighbor** {*ip-address* | *peer-group-name*} **ebgp-multihop** [*ttl*]
26. **neighbor** {*ip-address* | *peer-group-name*} **activate**
27. **neighbor** *ip-address* **allowas-in** [*number*]
28. **no auto-summary**
29. **no synchronization**
30. Repeat Step 20 to Step 29 to configure another VRF instance.
31. **end**
32. **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*}

DETAILED STEPS

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

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface loopback0`	Configures an interface type and enters interface configuration mode. • In this example, loopback interface 0 is configured.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 172.16.1.1 255.255.255.255`	Configures an IP address. • In this example, the loopback interface is configured with an IP address of 172.16.1.1.
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**interface** *type number* **Example:** `Router(config)# interface loopback1`	Configures an interface type and enters interface configuration mode. • In this example, loopback interface 1 is configured.
Step 7	**ip vrf forwarding** *vrf-name* [**downstream** *vrf-name2*] **Example:** `Router(config-if)# ip vrf forwarding vrf_trans`	Associates a VRF with an interface or subinterface. • In this example, the VRF named vrf_trans is associated with loopback interface 1. **Note** Executing this command on an interface removes the IP address. The IP address should be reconfigured.
Step 8	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 10.99.1.1 255.255.255.255`	Configures an IP address. • In this example, loopback interface 1 is configured with an IP address of 10.99.1.1.
Step 9	Repeat Step 5 through Step 8 for each VRF to be associated with an interface.	--
Step 10	**exit** **Example:**	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
	`Router(config-if)# exit`	
Step 11	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 12	**bgp router-id** {*ip-address* | **vrf auto-assign**} **Example:** `Router(config-router)# bgp router-id vrf auto-assign`	Configures a fixed router ID for the local BGP routing process. • In this example, a BGP router ID is automatically assigned for each VRF instance.
Step 13	**no bgp default ipv4-unicast** **Example:** `Router(config-router)# no bgp default ipv4-unicast`	Disables the IPv4 unicast address family for the BGP routing process. **Note** Routing information for the IPv4 unicast address family is advertised by default for each BGP routing session configured with the **neighbor remote-as** router configuration command unless you configure the **no bgp default ipv4-unicast** router configuration command before configuring the **neighbor remote-as** command. Existing neighbor configurations are not affected.
Step 14	**bgp log-neighbor-changes** **Example:** `Router(config-router)# bgp log-neighbor-changes`	Enables logging of BGP neighbor resets.
Step 15	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor is an internal neighbor.
Step 16	**neighbor** {*ip-address* | *peer-group-name*} **update-source** *interface-type interface-number*	Allows BGP sessions to use any operational interface for TCP connections.

	Command or Action	Purpose
	Example: `Router(config-router)# neighbor 192.168.1.1 update-source loopback0`	• In this example, BGP TCP connections for the specified neighbor are sourced with the IP address of the loopback interface rather than the best local address.
Step 17	**address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]} **Example:** `Router(config-router)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations. • The example creates a VPNv4 address family session.
Step 18	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Router(config-router-af)# neighbor 172.16.1.1 activate`	Activates the neighbor under the VPNv4 address family. • In this example, the neighbor 172.16.1.1 is activated.
Step 19	**neighbor** {*ip-address* | *peer-group-name*} **send-community** {**both** | **standard** | **extended**} **Example:** `Router(config-router-af)# neighbor 172.16.1.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • In this example, an extended communities attribute is sent to the neighbor at 172.16.1.1.
Step 20	**exit-address-family** **Example:** `Router(config-router-af)# exit-address-family`	Exits address family configuration mode and returns to router configuration mode.
Step 21	**address-family** {**ipv4** [**mdt** | **multicast** | **unicast** [**vrf** *vrf-name*] | **vrf** *vrf-name*] | **vpnv4** [**unicast**]} **Example:** `Router(config-router)# address-family ipv4 vrf vrf_trans`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations. • The example specifies that the VRF instance named vrf_trans is to be associated with subsequent IPv4 address family configuration mode commands.
Step 22	**redistribute connected** **Example:** `Router(config-router-af)# redistribute connected`	Redistributes from one routing domain into another routing domain. • In this example, the **connected** keyword is used to represent routes that are established automatically when IP is enabled on an interface. • Only the syntax applicable to this step is displayed. For more details, see the *Cisco IOS IP Routing: BGP Command Reference*.

	Command or Action	Purpose
Step 23	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.1.1 remote-as 40000`	Adds the IP address or peer group name of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • If the *autonomous-system-number* argument matches the autonomous system number specified in the **router bgp** command, the neighbor is an internal neighbor. • If the *autonomous-system-number* argument does not match the autonomous system number specified in the **router bgp** command, the neighbor is an external neighbor. • In this example, the neighbor at 192.168.1.1 is an external neighbor.
Step 24	**neighbor** *ip-address* **local-as** *autonomous-system-number* [**no-prepend** [**replace-as** [**dual-as**]]] **Example:** `Router(config-router-af)# neighbor 192.168.1.1 local-as 50000 no-prepend`	Customizes the AS_PATH attribute for routes received from an eBGP neighbor. • The autonomous system number from the local BGP routing process is prepended to all external routes by default. • Use the **no-prepend** keyword to not prepend the local autonomous system number to any routes received from the eBGP neighbor. • In this example, routes from the neighbor at 192.168.1.1 will not contain the local autonomous system number.
Step 25	**neighbor** {*ip-address* | *peer-group-name*} **ebgp-multihop** [*ttl*] **Example:** `Router(config-router-af)# neighbor 192.168.1.1 ebgp-multihop 2`	Accepts and attempts BGP connections to external peers residing on networks that are not directly connected. • In this example, BGP is configured to allow connections to or from neighbor 192.168.1.1, which resides on a network that is not directly connected.
Step 26	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Router(config-router-af)# neighbor 192.168.1.1 activate`	Activates the neighbor under the IPV4 address family. • In this example, the neighbor 192.168.1.1 is activated.
Step 27	**neighbor** *ip-address* **allowas-in** [*number*] **Example:** `Router(config-router-af)# neighbor 192.168.1.1 allowas-in 1`	Configures provider edge (PE) routers to allow the readvertisement of all prefixes that contain duplicate autonomous system numbers. • In the example, the PE router with autonomous system number 45000 is configured to allow prefixes from the VRF vrf-trans. The neighboring PE router with

	Command or Action	Purpose
		the IP address 192.168.1.1 is set to be readvertised once to other PE routers with the same autonomous system number.
Step 28	**no auto-summary** **Example:** `Router(config-router-af)# no auto-summary`	Disables automatic summarization and sends subprefix routing information across classful network boundaries.
Step 29	**no synchronization** **Example:** `Router(config-router-af)# no synchronization`	Enables the Cisco IOS software to advertise a network route without waiting for synchronization with an Internal Gateway Protocol (IGP).
Step 30	Repeat Step 20 to Step 29 to configure another VRF instance.	--
Step 31	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 32	**show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} **Example:** `Router# show ip bgp vpnv4 all`	(Optional) Displays VPN address information from the BGP table. • In this example, the complete VPNv4 database is displayed. **Note** Only the syntax applicable to this task is used in this example. For more details, see the *Cisco IOS Multiprotocol Label Switching Command Reference*.

Examples

The following sample output assumes that two VRF instances named vrf_trans and vrf_user were configured, each with a separate router ID. The router ID is shown next to the VRF name.

```
Router# show ip bgp vpnv4 all

BGP table version is 43, local router ID is 172.16.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf vrf_trans) VRF Router ID 10.99.1.2
*> 172.22.0.0       0.0.0.0                  0         32768 ?
r> 172.23.0.0       172.23.1.1               0             0 3 1 ?
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
*> 10.52.1.0/24     172.23.1.1                             0 3 1 ?
```

```
*> 10.52.2.1/32     172.23.1.1                            0 3 1 3 i
*> 10.52.3.1/32     172.23.1.1                            0 3 1 3 i
*> 10.99.1.1/32     172.23.1.1               0            0 3 1 ?
*> 10.99.1.2/32     0.0.0.0                  0        32768 ?
Route Distinguisher: 10:1
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
Route Distinguisher: 42:1 (default for vrf vrf_user) VRF Router ID 10.99.1.1
r> 172.22.0.0       172.22.1.1               0            0 2 1 ?
*> 172.23.0.0       0.0.0.0                  0        32768 ?
*> 10.21.1.1/32     172.22.1.1                            0 2 1 2 i
*>i10.52.1.0/24     192.168.3.1              0    100      0 ?
*>i10.52.2.1/32     192.168.3.1              0    100      0 3 i
*>i10.52.3.1/32     192.168.3.1              0    100      0 3 i
*> 10.99.1.1/32     0.0.0.0                  0        32768 ?
*> 10.99.1.2/32     172.22.1.1               0            0 2 1 ?
```

Configuration Examples for Per-VRF Assignment of BGP Router ID

Example: Manually Configuring a BGP Router ID per VRF

The following example shows how to configure two VRFs—vrf_trans and vrf_user—with sessions between each other on the same router. The BGP router ID for each VRF is configured manually under separate IPv4 address families. The **show ip bgp vpnv4** command can be used to verify that the router IDs have been configured for each VRF. The configuration starts in global configuration mode.

```
ip vrf vrf_trans
 rd 45000:1
 route-target export 50000:50
 route-target import 40000:1
!
ip vrf vrf_user
 rd 65500:1
 route-target export 65500:1
 route-target import 65500:1
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
interface Ethernet0/0
 ip vrf forwarding vrf_trans
 ip address 172.22.1.1 255.255.0.0
!
interface Ethernet1/0
 ip vrf forwarding vrf_user
 ip address 172.23.1.1 255.255.0.0
!
router bgp 45000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 192.168.3.1 remote-as 45000
 neighbor 192.168.3.1 update-source Loopback0
 !
 address-family vpnv4
  neighbor 192.168.3.1 activate
  neighbor 192.168.3.1 send-community extended
```

```
  exit-address-family
 !
 address-family ipv4 vrf vrf_user
  redistribute connected
  neighbor 172.22.1.1 remote-as 40000
  neighbor 172.22.1.1 local-as 50000 no-prepend
  neighbor 172.22.1.1 ebgp-multihop 2
  neighbor 172.22.1.1 activate
  neighbor 172.22.1.1 allowas-in 1
  no auto-summary
  no synchronization
  bgp router-id 10.99.1.1
  exit-address-family
 !
 address-family ipv4 vrf vrf_trans
  redistribute connected
  neighbor 172.23.1.1 remote-as 50000
  neighbor 172.23.1.1 local-as 40000 no-prepend
  neighbor 172.23.1.1 ebgp-multihop 2
  neighbor 172.23.1.1 activate
  neighbor 172.23.1.1 allowas-in 1
  no auto-summary
  no synchronization
  bgp router-id 10.99.1.2
  exit-address-family
```

After the configuration, the output of the **show ip bgp vpnv4 all** command shows the router ID displayed next to the VRF name:

```
Router# show ip bgp vpnv4 all

BGP table version is 43, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 45000:1 (default for vrf vrf_trans) VRF Router ID 10.99.1.2
*> 172.22.0.0       0.0.0.0                  0         32768 ?
r> 172.23.0.0       172.23.1.1               0             0 3 1 ?
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
*> 10.52.1.0/24     172.23.1.1                             0 3 1 ?
*> 10.52.2.1/32     172.23.1.1                             0 3 1 3 i
*> 10.52.3.1/32     172.23.1.1                             0 3 1 3 i
*> 10.99.1.1/32     172.23.1.1               0             0 3 1 ?
*> 10.99.2.2/32     0.0.0.0                  0         32768 ?
Route Distinguisher: 50000:1
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
Route Distinguisher: 65500:1 (default for vrf vrf_user) VRF Router ID 10.99.1.1
r> 172.22.0.0       172.22.1.1               0             0 2 1 ?
*> 172.23.0.0       0.0.0.0                  0         32768 ?
*> 10.21.1.1/32     172.22.1.1                             0 2 1 2 i
*>i10.52.1.0/24     192.168.3.1              0    100      0 ?
*>i10.52.2.1/32     192.168.3.1              0    100      0 3 i
*>i10.52.3.1/32     192.168.3.1              0    100      0 3 i
*> 10.99.1.1/32     0.0.0.0                  0         32768 ?
*> 10.99.2.2/32     172.22.1.1               0             0 2 1 ?
```

The output of the **show ip bgp vpnv4 vrf** command for a specified VRF displays the router ID in the output header:

```
Router# show ip bgp vpnv4 vrf vrf_user

BGP table version is 43, local router ID is 10.99.1.1
```

```
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 65500:1 (default for vrf vrf_user) VRF Router ID 10.99.1.1
r> 172.22.0.0       172.22.1.1               0             0 2 1 ?
*> 172.23.0.0       0.0.0.0                  0         32768 ?
*> 10.21.1.1/32     172.22.1.1                             0 2 1 2 i
*>i10.52.1.0/24     192.168.3.1              0    100      0 ?
*>i10.52.2.1/32     192.168.3.1              0    100      0 3 i
*>i10.52.3.1/32     192.168.3.1              0    100      0 3 i
*> 10.99.1.1/32     0.0.0.0                  0         32768 ?
*> 10.99.2.2/32     172.22.1.1               0             0 2 1 ?
```

The output of the **show ip bgp vpnv4 vrf summary** command for a specified VRF displays the router ID in the first line of the output:

```
Router# show ip bgp vpnv4 vrf vrf_user summary

BGP router identifier 10.99.1.1, local AS number 45000
BGP table version is 43, main routing table version 43
8 network entries using 1128 bytes of memory
8 path entries using 544 bytes of memory
16/10 BGP path/bestpath attribute entries using 1856 bytes of memory
6 BGP AS-PATH entries using 144 bytes of memory
3 BGP extended community entries using 72 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 3744 total bytes of memory
BGP activity 17/0 prefixes, 17/0 paths, scan interval 15 secs
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
172.22.1.1      4     2      20      21       43    0    0 00:12:33        3
```

When the path is sourced in the VRF, the correct router ID is displayed in the output of the **show ip bgp vpnv4 vrf**command for a specified VRF and network address:

```
Router# show ip bgp vpnv4 vrf vrf_user 172.23.0.0

BGP routing table entry for 65500:1:172.23.0.0/8, version 22
Paths: (1 available, best #1, table vrf_user)
  Advertised to update-groups:
     2          3
  Local
    0.0.0.0 from 0.0.0.0 (10.99.1.1)
      Origin incomplete, metric 0, localpref 100, weight 32768, valid, sourced, best
      Extended Community: RT:65500:1
```

Example: Automatically Assigning a BGP Router ID per VRF

The following three examples show different methods of configuring BGP to automatically assign a separate router ID to each VRF instance.

Globally Automatically Assigned Router ID Using Loopback Interface IP Addresses

The following example shows how to configure two VRFs—vrf_trans and vrf_user—with sessions between each other on the same router. Under router configuration mode, BGP is globally configured to automatically assign each VRF a BGP router ID. Loopback interfaces are associated with individual VRFs to source an IP address for the router ID. The **show ip bgp vpnv4** command can be used to verify that the router IDs have been configured for each VRF.

```
ip vrf vrf_trans
 rd 45000:1
 route-target export 50000:50
 route-target import 40000:1
!
ip vrf vrf_user
 rd 65500:1
 route-target export 65500:1
 route-target import 65500:1
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
interface Loopback1
 ip vrf forwarding vrf_user
 ip address 10.99.1.1 255.255.255.255
!
interface Loopback2
 ip vrf forwarding vrf_trans
 ip address 10.99.2.2 255.255.255.255
!
interface Ethernet0/0
 ip vrf forwarding vrf_trans
 ip address 172.22.1.1 255.0.0.0
!
interface Ethernet1/0
 ip vrf forwarding vrf_user
 ip address 172.23.1.1 255.0.0.0
!
router bgp 45000
 bgp router-id vrf auto-assign
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 192.168.3.1 remote-as 45000
 neighbor 192.168.3.1 update-source Loopback0
 !
address-family vpnv4
  neighbor 192.168.3.1 activate
  neighbor 192.168.3.1 send-community extended
  exit-address-family
 !
 address-family ipv4 vrf vrf_user
  redistribute connected
  neighbor 172.22.1.1 remote-as 40000
  neighbor 172.22.1.1 local-as 50000 no-prepend
  neighbor 172.22.1.1 ebgp-multihop 2
  neighbor 172.22.1.1 activate
  neighbor 172.22.1.1 allowas-in 1
  no auto-summary
  no synchronization
  exit-address-family
 !
 address-family ipv4 vrf vrf_trans
  redistribute connected
  neighbor 172.23.1.1 remote-as 50000
  neighbor 172.23.1.1 local-as 2 no-prepend
  neighbor 172.23.1.1 ebgp-multihop 2
  neighbor 172.23.1.1 activate
  neighbor 172.23.1.1 allowas-in 1
  no auto-summary
  no synchronization
  exit-address-family
```

After the configuration, the output of the **show ip bgp vpnv4 all** command shows the router ID displayed next to the VRF name. Note that the router IDs used in this example are sourced from the IP addresses configured for loopback interface 1 and loopback interface 2. The router IDs are the same as in the “Example: Manually Configurinig a BGP Router ID per VRF” section.

```
Router# show ip bgp vpnv4 all

BGP table version is 43, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 45000:1 (default for vrf vrf_trans) VRF Router ID 10.99.2.2
*> 172.22.0.0       0.0.0.0                  0         32768 ?
r> 172.23.0.0       172.23.1.1               0             0 3 1 ?
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
*> 10.52.1.0/24     172.23.1.1                             0 3 1 ?
*> 10.52.2.1/32     172.23.1.1                             0 3 1 3 i
*> 10.52.3.1/32     172.23.1.1                             0 3 1 3 i
*> 10.99.1.1/32     172.23.1.1               0             0 3 1 ?
*> 10.99.1.2/32     0.0.0.0                  0         32768 ?
Route Distinguisher: 50000:1
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
Route Distinguisher: 65500:1 (default for vrf vrf_user) VRF Router ID 10.99.1.1
r> 172.22.0.0       172.22.1.1               0             0 2 1 ?
*> 172.23.0.0       0.0.0.0                  0         32768 ?
*> 10.21.1.1/32     172.22.1.1                             0 2 1 2 i
*>i10.52.1.0/24     192.168.3.1              0    100      0 ?
*>i10.52.2.1/32     192.168.3.1              0    100      0 3 i
*>i10.52.3.1/32     192.168.3.1              0    100      0 3 i
*> 10.99.1.1/32     0.0.0.0                  0         32768 ?
*> 10.99.1.2/32     172.22.1.1               0             0 2 1 ?
```

Globally Automatically Assigned Router ID with No Default Router ID

The following example shows how to configure a router and associate a VRF that is automatically assigned a BGP router ID when no default router ID is allocated.

```
ip vrf vpn1
 rd 45000:1
 route-target export 45000:1
 route-target import 45000:1
!
interface Loopback0
 ip vrf forwarding vpn1
 ip address 10.1.1.1 255.255.255.255
!
interface Ethernet0/0
 ip vrf forwarding vpn1
 ip address 172.22.1.1 255.0.0.0
!
router bgp 45000
 bgp router-id vrf auto-assign
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 !
 address-family ipv4 vrf vpn1
  neighbor 172.22.1.2 remote-as 40000
  neighbor 172.22.1.2 activate
  no auto-summary
```

```
  no synchronization
  exit-address-family
```

Assuming that a second router is configured to establish a session between the two routers, the output of the **show ip interface brief** command shows only the VRF interfaces that are configured.

```
Router# show ip interface brief

Interface              IP-Address      OK? Method Status                Protocol
Ethernet0/0            172.22.1.1      YES NVRAM  up                    up
Ethernet1/0            unassigned      YES NVRAM  administratively down down
Serial2/0              unassigned      YES NVRAM  administratively down down
Serial3/0              unassigned      YES NVRAM  administratively down down
Loopback0              10.1.1.1        YES NVRAM  up                    up
```

The **show ip vrf** command can be used to verify that a router ID is assigned for the VRF:

```
Router# show ip vrf

Name                             Default RD          Interfaces
  vpn1                             45000:1             Loopback0
                                                       Ethernet0/0
VRF session is established:
```

Per-VRF Automatically Assigned Router ID

The following example shows how to configure two VRFs—vrf_trans and vrf_user—with sessions between each other on the same router. Under the IPv4 address family associated with an individual VRF, BGP is configured to automatically assign a BGP router ID. Loopback interfaces are associated with individual VRFs to source an IP address for the router ID. The output of the **show ip bgp vpnv4** command can be used to verify that the router IDs have been configured for each VRF.

```
ip vrf vrf_trans
 rd 45000:1
 route-target export 50000:50
 route-target import 40000:1
!
ip vrf vrf_user
 rd 65500:1
 route-target export 65500:1
 route-target import 65500:1
!
interface Loopback0
 ip address 10.1.1.1 255.255.255.255
!
interface Loopback1
 ip vrf forwarding vrf_user
 ip address 10.99.1.1 255.255.255.255
!
interface Loopback2
 ip vrf forwarding vrf_trans
 ip address 10.99.2.2 255.255.255.255
!
interface Ethernet0/0
 ip vrf forwarding vrf_trans
 ip address 172.22.1.1 255.0.0.0
!
interface Ethernet1/0
 ip vrf forwarding vrf_user
 ip address 172.23.1.1 255.0.0.0
!
```

```
router bgp 45000
 no bgp default ipv4-unicast
 bgp log-neighbor-changes
 neighbor 192.168.3.1 remote-as 45000
 neighbor 192.168.3.1 update-source Loopback0
 !
address-family vpnv4
  neighbor 192.168.3.1 activate
  neighbor 192.168.3.1 send-community extended
  exit-address-family
!
 address-family ipv4 vrf vrf_user
  redistribute connected
  neighbor 172.22.1.1 remote-as 40000
  neighbor 172.22.1.1 local-as 50000 no-prepend
  neighbor 172.22.1.1 ebgp-multihop 2
  neighbor 172.22.1.1 activate
  neighbor 172.22.1.1 allowas-in 1
  no auto-summary
  no synchronization
  bgp router-id auto-assign
  exit-address-family
 !
 address-family ipv4 vrf vrf_trans
  redistribute connected
  neighbor 172.23.1.1 remote-as 50000
  neighbor 172.23.1.1 local-as 40000 no-prepend
  neighbor 172.23.1.1 ebgp-multihop 2
  neighbor 172.23.1.1 activate
  neighbor 172.23.1.1 allowas-in 1
  no auto-summary
  no synchronization
  bgp router-id auto-assign
  exit-address-family
```

After the configuration, the output of the **show ip bgp vpnv4 all** command shows the router ID displayed next to the VRF name. Note that the router IDs used in this example are sourced from the IP addresses configured for loopback interface 1 and loopback interface 2.

```
Router# show ip bgp vpnv4 all

BGP table version is 43, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 45000:1 (default for vrf vrf_trans) VRF Router ID 10.99.2.2
*> 172.22.0.0       0.0.0.0                  0         32768 ?
r> 172.23.0.0       172.23.1.1               0             0 3 1 ?
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
*> 10.52.1.0/24     172.23.1.1                             0 3 1 ?
*> 10.52.2.1/32     172.23.1.1                             0 3 1 3 i
*> 10.52.3.1/32     172.23.1.1                             0 3 1 3 i
*> 10.99.1.1/32     172.23.1.1               0             0 3 1 ?
*> 10.99.1.2/32     0.0.0.0                  0         32768 ?
Route Distinguisher: 50000:1
*>i10.21.1.1/32     192.168.3.1              0    100      0 2 i
Route Distinguisher: 65500:1 (default for vrf vrf_user) VRF Router ID 10.99.1.1
r> 172.22.0.0       172.22.1.1               0             0 2 1 ?
*> 172.23.0.0       0.0.0.0                  0         32768 ?
*> 10.21.1.1/32     172.22.1.1                             0 2 1 2 i
*>i10.52.1.0/24     192.168.3.1              0    100      0 ?
*>i10.52.2.1/32     192.168.3.1              0    100      0 3 i
```

```
*>i10.52.3.1/32     192.168.3.1             0    100      0 3 i
*> 10.99.1.1/32     0.0.0.0                 0         32768 ?
*> 10.99.1.2/32     172.22.1.1              0             0 2 1 ?
```

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
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference

Standards

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

MIBs

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

RFCs

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

Technical Assistance

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

Feature Information for Per-VRF Assignment of BGP Router ID

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 49: Feature Information for Per-VRF Assignment of BGP Router ID

Feature Name	Releases	Feature Information
Per-VRF Assignment of BGP Router ID	12.2(31)SB2 12.2(33)SRA 12.2(33)SXH 12.4(20)T 15.0(1)S	The Per-VRF Assignment of BGP Router ID feature introduces the ability to have VRF-to-VRF peering in Border Gateway Protocol (BGP) on the same router. BGP is designed to refuse a session with itself because of the router ID check. The per-VRF assignment feature allows a separate router ID per VRF using a new keyword in the existing **bgp router-id** command. The router ID can be manually configured for each VRF or can be assigned automatically either globally under address family configuration mode or for each VRF. The following commands were introduced or modified by this feature: **bgp router-id**, **show ip bgp vpnv4**.

CHAPTER 37

BGP Event-Based VPN Import

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

Finding Feature Information

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

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

Prerequisites for BGP Event-Based VPN Import

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

Information About BGP Event-Based VPN Import

BGP Event-Based VPN Import

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

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

Import Path Selection Policy

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

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

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

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

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

Import Path Limit

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

How to Configure BGP Event-Based VPN Import

Configuring a Multiprotocol VRF

Perform this task to configure a multiprotocol VRF that allows you to share route-target policies (import and export) between IPv4 and IPv6 or to configure separate route-target policies for IPv4 and IPv6 VPNs. In this task, only the IPv4 address family is configured, but we recommend using the multiprotocol VRF configuration for all new VRF configurations.

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**vrf definition** *vrf-name* **Example:** `Router(config)# vrf definition vrf-A`	Configures a VRF routing table and enters VRF configuration mode. • Use the *vrf-name* argument to specify a name to be assigned to the VRF.
Step 4	**rd** *route-distinguisher* **Example:** `Router(config-vrf)# rd 45000:1`	Creates routing and forwarding tables and specifies the default route distinguisher for a VPN. • Use the *route-distinguisher* argument to add an 8-byte value to an IPv4 prefix to create a unique VPN IPv4 prefix.
Step 5	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Router(config-vrf)# route-target both 45000:100`	Creates a route target extended community for a VRF. • Use the **import** keyword to import routing information from the target VPN extended community. • Use the **export** keyword to export routing information to the target VPN extended community. • Use the **both** keyword to both import routing information from, and export routing information to, the target VPN extended community. • Use the *route-target-ext-community* argument to add the route target extended community attributes to the VRF's list of import, export, or both (import and export) route target extended communities.
Step 6	**address-family ipv4** [**unicast**] **Example:** `Router(config-vrf)# address-family ipv4 unicast`	Specifies the IPv4 address family and enters VRF address family configuration mode. • This step is required here to specify an address family for the VRF defined in the previous steps.
Step 7	**exit-address-family** **Example:** `Router(config-vrf-af)# exit-address-family`	Exits VRF address family configuration mode and returns to VRF configuration mode.
Step 8	**exit** **Example:** `Router(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.

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

Configuring Event-Based VPN Import Processing for BGP Paths

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

Before you begin

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv4 vrf** *vrf-name* **Example:** `Router(config-router)# address-family ipv4 vrf vrf-A`	Specifies the IPv4 address family and enters address family configuration mode. • Use the **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**import path selection** {**all**	**bestpath** [**strict**]	**multipath** [**strict**]} **Example:** `Router(config-router-af)# import path selection all`	Specifies the BGP path selection policy for importing routes into a VRF table. • In this example, all paths that match any RT of the VRF instance are imported.
Step 6	**import path limit** *number-of-import-paths* **Example:** `Router(config-router-af)# import path limit 3`	Specifies, per importing net, a maximum number of BGP paths that can be imported from an exporting net.

	Command or Action	Purpose
Step 7	**end** **Example:** `Router(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Monitoring and Troubleshooting BGP Event-Based VPN Import Processing

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

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

SUMMARY STEPS

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

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 **show ip bgp vpnv4** {**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} [*network-address* [*mask*]]

In this example output, a safe import path selection policy is in effect because the **strict** keyword is not configured using the **import path selection** command. When a path is imported as the best available path (when the bestpath or multipaths are not eligible for import), the path is marked with "imported safety path," as shown in the output.

Example:

```
Router# show ip bgp vpnv4 all 172.17.0.0

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

The paths existing in an exporting net that are considered for import into a VRF instance may have been received from another peer router and were not subject to the VPN importing rules. These paths may contain the same route-distinguisher

(RD) information because the RD information is local to a router, but some of these paths do not match the RT of the importing VRF instance and are marked as "not-in-vrf" in the **show** command output.

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

Example:

```
Router# show ip bgp vpnv4 all 172.17.0.0

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

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

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

Example:

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

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

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

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

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

Example:

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

```
BGP import events debugging is on
```

Configuration Examples for BGP Event-Based VPN Import

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

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

```
vrf definition vrf-A
 rd 45000:1
 route-target import 45000:100
 address-family ipv4
  exit-address-family
!
interface FastEthernet1/1
 no ip address
 vrf forwarding vrf-A
 ip address 10.4.8.149 255.255.255.0
 no shut
 exit
!
router bgp 45000
 network 172.17.1.0 mask 255.255.255.0
 neighbor 192.168.1.2 remote-as 40000
 neighbor 192.168.3.2 remote-as 50000
 address-family ipv4 vrf vrf-A
  import path selection all
  import path limit 3
  exit-address-family
 address-family vpnv4
  neighbor 192.168.1.2 activate
  neighbor 192.168.3.2 activate
  end
```

Additional References

Related Documents

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

Standards and RFCs

Standard/RFC	Title
RFC 2918	*Route Refresh Capability for BGP-4*

Technical Assistance

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

Feature Information for BGP Event-Based VPN Import

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

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

Table 50: Feature Information for BGP Event-Based VPN Import

Feature Name	Releases	Feature Information
BGP Event-Based VPN Import	15.0(1)M	The BGP Event-Based VPN Import feature introduces a modification to the existing Border Gateway Protocol (BGP) path import process. The enhanced BGP path import is driven by events; when a BGP path changes, all of its imported copies are updated as soon as processing is available. Convergence times are significantly reduced because there is no longer any delay in the propagation of routes due to the software waiting for a periodic scanner time interval before processing the updates. To implement the new processing, new command-line interface (CLI) commands are introduced. The following commands were introduced or modified: **bgp scan-time**, **import path limit**, **import path selection**, **maximum-paths eibgp**, **maximum-paths ibgp**, **show ip bgp vpnv4**.

CHAPTER 38

Configuring BGP: RT Constrained Route Distribution

BGP: RT Constrained Route Distribution is a feature that can be used by service providers in Multiprotocol Label Switching (MPLS) Layer 3 VPNs to reduce the number of unnecessary routing updates that route reflectors (RRs) send to Provider Edge (PE) routers. The reduction in routing updates saves resources by allowing RRs, Autonomous System Boundary Routers (ASBRs), and PEs to have fewer routes to carry. Route targets are used to constrain routing 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 at the end of this module.

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

Prerequisites for BGP: RT Constrained Route Distribution

Before you configure BGP: RT Constrained Route Distribution, you should understand how to configure the following:

- Multiprotocol Label Switching (MPLS) VPNs
- Route distinguishers (RDs)

- Route targets (RTs)
- Multiprotocol BGP (MBGP)

Restrictions for BGP: RT Constrained Route Distribution

BGP: RT Constrained Route Distribution constrains all VPN route advertisements.

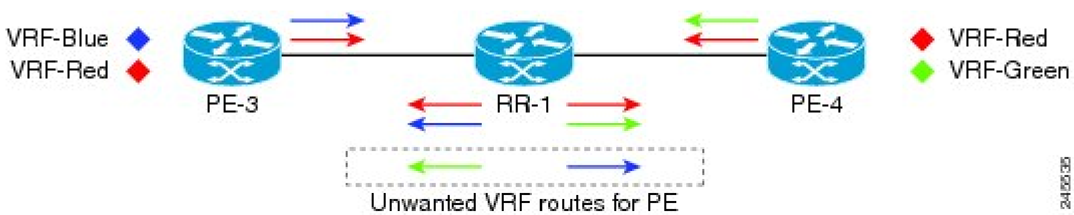
Information About BGP: RT Constrained Route Distribution

Problem That BGP: RT Constrained Route Distribution Solves

Some service providers have a large number of routing updates being sent from RRs to PEs, which can require extensive use of resources. A PE does not need routing updates for VRFs that are not on the PE; therefore, the PE determines that many routing updates it receives are "unwanted." The PE filters out the unwanted updates.

The figure below illustrates a scenario in which unwanted routing updates arrive at two PEs.

Figure 54: Unwanted Routing Updates at PE

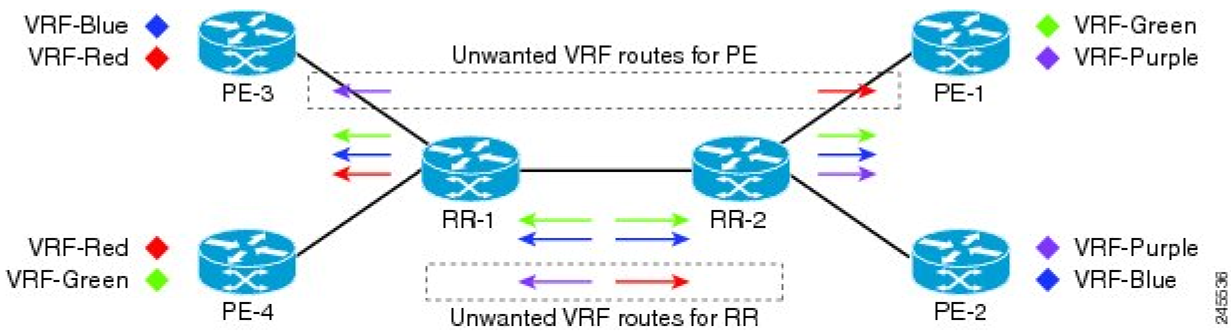


As shown in the figure above, a PE receives unwanted routes in the following manner:

1. PE-3 advertises VRF Blue and VRF Red routes to RR-1. PE-4 advertises VRF Red and VRF Green routes to RR-1.
2. RR-1 has all of the routes for all of the VRFs (Blue, Red, and Green).
3. During a route refresh or VRF provisioning, RR-1 advertises all of the VRF routes to both PE-3 and PE-4.
4. Routes for VRF Green are unwanted at PE-3. Routes for VRF Blue are unwanted at PE-4.

Now consider the scenario where there are two RRs with another set of PEs. There are unwanted routing updates from RRs to PEs and unwanted routing updates between RRs. The figure below illustrates a scenario in which unwanted routes arrive at an RR.

Figure 55: Unwanted Routing Updates at RR



As shown in the figure above, RR-1 and RR-2 receive unwanted routing updates in the following manner:

1. PE-3 and PE-4 advertise VRF Blue, VRF Red, and VRF Green VPN routes to RR-1.
2. RR-1 sends all of its VPN routes to RR-2.
3. VRF Red routes are unwanted on RR-2 because PE-1 and PE-2 do not have VRF Red.
4. Similarly, VRF Purple routes are unwanted on RR-1 because PE-3 and PE-4 do not have VRF Purple.

Hence, a large number of unwanted routes might be advertised among RRs and PEs. The BGP: RT Constrained Route Distribution feature addresses this problem by filtering unwanted routing updates.

Before the BGP: RT Constrained Route Distribution feature, the PE would filter the updates. With this feature, the burden is moved to the RR to filter the updates.

Benefits of BGP: RT Constrained Route Distribution

In MPLS L3VPNs, PE routers use BGP and route target (RT) extended communities to control the distribution of VPN routes to and from VRFs in order to separate the VPNs. PEs and Autonomous System Boundary Routers (ASBRs) commonly receive and then filter out the unwanted VPN routes.

However, receiving and filtering unwanted VPN routes is a waste of resources. The sender generates and transmits a VPN routing update and the receiver filters out the unwanted routes. Preventing the generation of VPN route updates would save resources.

Route Target Constrain (RTC) is a mechanism that prevents the propagation of VPN Network Layer Reachability Information (NLRI) from the RR to a PE that is not interested in the VPN. The feature provides considerable savings in CPU cycles and transient memory usage. RT constraint limits the number of VPN routes and describes VPN membership.

BGP RT-Constrain SAFI

The BGP: RT Constrained Route Distribution feature introduces the BGP RT-Constrain Subsequent Address Family Identifier (SAFI). The command to enter that address family is the **address-family rtfilter unicast** command.

BGP: RT Constrained Route Distribution Operation

In order to filter out the unwanted routes described in the "Problem that BGP RT Constrained Route Distribution Solves" section on page 2, the PEs and RRs must be configured with the BGP: RT Constrained Route Distribution feature.

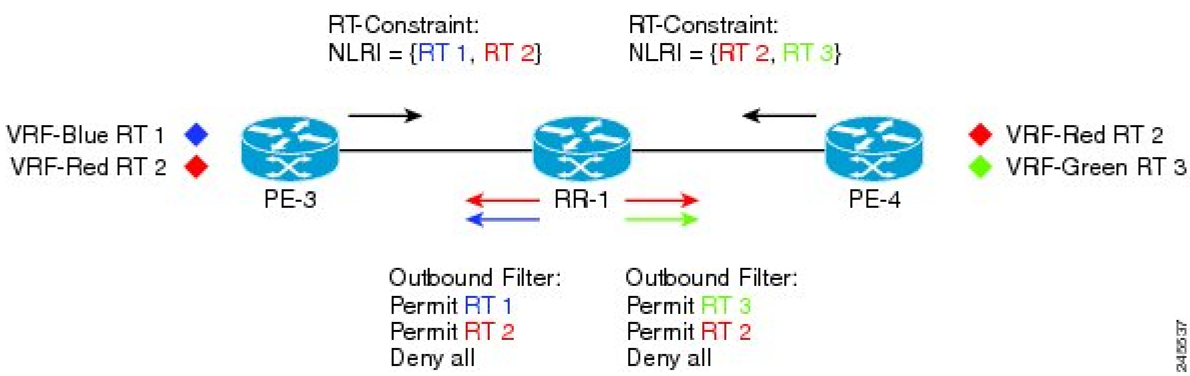
The feature allows the PE to propagate RT membership and use the RT membership to limit the VPN routing information maintained at the PE and RR. The PE uses an MP-BGP UPDATE message to propagate the membership information. The RR restricts advertisement of VPN routes based on the RT membership information it received.

This feature causes two exchanges to happen:

- The PE sends RT Constraint (RTC) Network Layer Reachability Information (NLRI) to the RR.
- The RR installs an outbound route filter.

The figure below illustrates the exchange of the RTC NLRI and the outbound route filter.

Figure 56: Exchange of RTC NLRI and Filter Between PE and RR



As shown in the figure above, the following exchange occurs between the PE and the RR:

1. PE-3 sends RTC NLRI (RT 1, RT 2) to RR-1.
2. PE-4 sends RTC NLRI (RT 2, RT 3) to RR-1.
3. RR-1 translates the NLRI into an outbound route filter and installs this filter (Permit RT 1, RT 2) for PE-3.
4. RR-1 translates the NLRI into an outbound route filter and installs this filter (Permit RT 2, RT 3) for PE-4.

RT Constraint NLRI Prefix

The format of the RT Constraint NLRI is a prefix that is always 12 bytes long, consisting of the following:

- 4-byte origin autonomous system
- 8-byte RT extended community value

The following are examples of RT Constraint prefixes:

- 65000:2:100:1
  - Origin autonomous system number is 65000

    - BGP Extended Community Type Code is 2
    - Route target is 100:1
- 65001:256:192.0.0.1:100
    - Origin ASN is 65001
    - BGP Extended Community Type Code is 256
    - Route target is 192.0.0.1:100
- 1.10:512:1.10:2
    - Origin ASN is 4-byte, unique 1.10
    - BGP Extended Community Type Code is 512
    - Route target is 1.10:2

To determine what the BGP Extended Community Type Code means, refer to RFC 4360, *BGP Extended Communities Attribute*. In the first example shown, a 2 translates in hexadecimal to 0x002. In RFC 4360, 0x002 indicates that the value that follows the type code will be a two-octet AS specific route target.

RT Constrained Route Distribution Process

This section shows the RT Constrained Route Distribution process. In this example has two CE routers in AS 100 that are connected to PE1. PE1 communicates with PE2, which is also connected to CE routers. Between the two PEs is a route reflector (RR). PE1 and PE2 belong to AS 65000.

The general process for the feature is as follows:

1. The user configures PE1 to activate its BGP peers under the **address-family rtfilter unicast** command.
2. The user configures PE1 in AS 65000 with **route-target import 100:1**, for example.
3. PE1 translates that command to an RT prefix of 65000:2:100:1. The 65000 is the service provider's AS number; the 2 is the BGP Extended Communities Type Code; and the 100:1 is the CE's RT (AS number and another number).
4. PE1 advertises the RT Constrain (RTC) prefix of 65000:2:100:1 to its iBGP peer RR.
5. The RR installs RTC 65000:2:100:1 into the RTC RIB. Each VRF has its own RIB.
6. The RR also installs RTC 65000:2:100:1 into its outbound filter for the neighbor PE1.
7. A filter in the RR either permits or denies the RT. (The AS number is ignored because iBGP is operating in a single AS and does not need to track the AS number.)
8. The RR looks in its outbound filter and sees that it permits outbound VPN packets for RT 100:1 to PE1. So, the RR sends VPN update packet only with RT 100:1 to PE1 and denies VPN updates with any other RT.

Default RT Filter

The default RT filter has a value of zero and length of zero. The default RT filter is used:

- By a peer to indicate that the peer wants all of the VPN routes sent to it, regardless of the RT value.
- By the RR to request that the PE advertise all of its VPN routes to the RR.

The default RT filter is created by configuring the **neighbor default-originate** command under the **address-family rtfilter unicast** command. On the RR it comes as default along with the configuration of route-reflector-client under the address-family rtfilter.

How to Configure RT Constrained Route Distribution

Configuring Multiprotocol BGP on Provider Edge (PE) Routers and Route Reflectors

Perform this task to configure multiprotocol BGP (MP-BGP) connectivity on the PE routers and route reflectors.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp default ipv4-unicast**
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number*
6. **address-family vpnv4** [**unicast**]
7. **neighbor** {*ip-address* | *peer-group-name*} **send-community extended**
8. **neighbor** {*ip-address* | *peer-group-name*} **activate**
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.

	Command or Action	Purpose
Step 4	**no bgp default ipv4-unicast** **Example:** `Device(config-router)# no bgp default ipv4-unicast`	(Optional) Disables the IPv4 unicast address family on all neighbors. • Use the **no** form of the **bgp default ipv4-unicast** command if you are using this neighbor for MPLS routes only.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor pp.0.0.1 remote-as 100`	Adds an entry to the BGP or multiprotocol BGP neighbor table. • The *ip-address* argument specifies the IP address of the neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 6	**address-family vpnv4** [**unicast**] **Example:** `Device(config-router)# address-family vpnv4`	Enters address family configuration mode for configuring routing sessions, such as BGP, that use standard VPNv4 address prefixes. • The optional **unicast** keyword specifies VPNv4 unicast address prefixes.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **send-community extended** **Example:** `Device(config-router-af)# neighbor pp.0.0.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • The *ip-address* argument specifies the IP address of the BGP-speaking neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group.
Step 8	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor pp.0.0.1 activate`	Enables the exchange of information with a neighboring BGP router. • The *ip-address* argument specifies the IP address of the neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group.
Step 9	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits to privileged EXEC mode.

Troubleshooting Tips

You can enter a **show ip bgp neighbor** command to verify that the neighbors are up and running. If this command is not successful, enter a **debug ip bgp** *ip-address* **events** command, where *ip-address* is the IP address of the neighbor.

Connecting the MPLS VPN Customers

To connect the MPLS VPN customers to the VPN, perform the following tasks:

Defining VRFs on PE Routers to Enable Customer Connectivity

To define virtual routing and forwarding (VRF) instances, perform this task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip vrf** *vrf-name*
4. **rd** *route-distinguisher*
5. **route-target** {**import** | **export** | **both**} *route-target-ext-community*
6. **import map** *route-map*
7. **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	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf vpn1`	Defines the VPN routing instance by assigning a VRF name and enters VRF configuration mode. • The *vrf-name* argument is the name assigned to a VRF.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:1`	Creates routing and forwarding tables. • The *route-distinguisher* argument adds an 8-byte value to an IPv4 prefix to create a VPN IPv4 prefix. You can enter an RD in either of these formats: • 16-bit AS number: your 32-bit number, for example, 101:3

	Command or Action	Purpose
		• 32-bit IP address: your 16-bit number, for example, 192.168.122.15:1
Step 5	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 100:1`	Creates a route-target extended community for a VRF. • The **import** keyword imports routing information from the target VPN extended community. • The **export** keyword exports routing information to the target VPN extended community. • The **both** keyword imports routing information from and exports routing information to the target VPN extended community. • The *route-target-ext-community* argument adds the RT extended community attributes to the VRF's list of import, export, or both (import and export) RT extended communities.
Step 6	**import map** *route-map* **Example:** `Device(config-vrf)# import map vpn1-route-map`	(Optional) Configures an import route map for a VRF. • The *route-map* argument specifies the route map to be used as an import route map for the VRF.
Step 7	**exit** **Example:** `Device(config-vrf)# exit`	(Optional) Exits to global configuration mode.

Configuring VRF Interfaces on PE Routers for Each VPN Customer

To associate a VRF with an interface or subinterface on the PE routers, perform this task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip vrf forwarding** *vrf-name*
5. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface Ethernet 5/0`	Specifies the interface to configure and enters interface configuration mode. • The *type* argument specifies the type of interface to be configured. • The *number* argument specifies the port, connector, or interface card number.
Step 4	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding vpn1`	Associates a VRF with the specified interface or subinterface. • The *vrf-name* argument is the name assigned to a VRF.
Step 5	**end** **Example:** `Device(config-if)# end`	(Optional) Exits to privileged EXEC mode.

Configuring BGP as the Routing Protocol Between the PE and CE Routers

To configure PE-to-CE routing sessions using BGP, perform this task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4** [**multicast** | **unicast** | **vrf** *vrf-name*]
5. **neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number*
6. **neighbor** {*ip-address* | *peer-group-name*} **activate**
7. **exit-address-family**
8. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**address-family ipv4** [**multicast** | **unicast** | **vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 vrf vpn1`	Specifies the IPv4 address family type and enters address family configuration mode. • The **multicast** keyword specifies IPv4 multicast address prefixes. • The **unicast** keyword specifies IPv4 unicast address prefixes. • The **vrf** *vrf-name* keyword and argument specify the name of the VRF to associate with subsequent IPv4 address family configuration mode commands.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **remote-as** *as-number* **Example:** `Device(config-router-af)# neighbor pp.0.0.1 remote-as 200`	Adds an entry to the BGP or multiprotocol BGP neighbor table. • The *ip-address* argument specifies the IP address of the neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group. • The *as-number* argument specifies the autonomous system to which the neighbor belongs.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor pp.0.0.1 activate`	Enables the exchange of information with a neighboring BGP router. • The *ip-address* argument specifies the IP address of the neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group.

	Command or Action	Purpose
Step 7	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode.
Step 8	**end** **Example:** `Device(config-router)# end`	(Optional) Exits to privileged EXEC mode.

Configuring RT Constraint on the PE

Perform this task on the PE to configure BGP: RT Constrained Route Distribution with the specified neighbor, and optionally verify that route target (RT) filtering is occurring.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family rtfilter unicast**
5. **neighbor** {*ip-address* | *peer-group-name*} **activate**
6. **neighbor** {*ip-address* | *peer-group-name*} **send-community extended**
7. **end**
8. **show ip bgp rtfilter all**
9. **show ip bgp rtfilter all summary**
10. **show ip bgp vpnv4 all**

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 1`	Configures a BGP routing process and enters router configuration mode.

	Command or Action	Purpose
Step 4	**address-family rtfilter unicast** **Example:** `Device(config-router)# address-family rtfilter unicast`	Specifies the RT filter address family type and enters address family configuration mode.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.0.0.1 activate`	Enables the exchange of automated RT filter information with the specified BGP neighbor.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **send-community extended** **Example:** `Device(config-router-af)# neighbor pp.0.0.1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • The *ip-address* argument specifies the IP address of the BGP-speaking neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group.
Step 7	**end** **Example:** `Device(config-router-af)# end`	Exits configuration mode and returns to privileged EXEC mode.
Step 8	**show ip bgp rtfilter all** **Example:** `Device# show ip bgp rtfilter all`	(Optional) Displays all BGP RT filter information.
Step 9	**show ip bgp rtfilter all summary** **Example:** `Device# show ip bgp rtfilter all summary`	(Optional) Displays summary BGP RT filter information.
Step 10	**show ip bgp vpnv4 all** **Example:** `Device# show ip bgp vpnv4 all`	(Optional) Displays summary BGP VPNv4 information.

Configuring RT Constraint on the RR

Perform this task on the RR to configure BGP: RT Constrained Route Distribution with the specified neighbor, and optionally verify that route target (RT) filtering is occurring.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family rtfilter unicast**
5. **neighbor** {*ip-address* | *peer-group-name*} **activate**
6. **neighbor** {*ip-address* | *peer-group-name*} **route-reflector-client**
7. **neighbor** {*ip-address* | *peer-group-name*} **send-community extended**
8. **end**
9. **show ip bgp rtfilter all**
10. **show ip bgp rtfilter all summary**
11. **show ip bgp vpnv4 all**

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 1`	Configures a BGP routing process and enters router configuration mode.
Step 4	**address-family rtfilter unicast** **Example:** `Device(config-router)# address-family rtfilter unicast`	Specifies the RT filter address family type and enters address family configuration mode.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.0.0.2 activate`	Enables RT Constraint with the specified BGP neighbor.
Step 6	**neighbor** {*ip-address* | *peer-group-name*} **route-reflector-client** **Example:**	Enables route-reflector-client funtionality under RT Constraint with the specified BGP neighbor. • Note that the route-reflector-client under RT Constraint address-family comes with a default

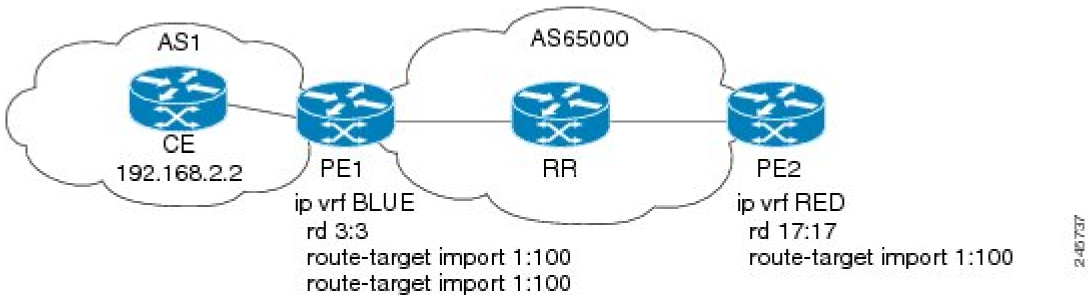
	Command or Action	Purpose
	`Device(config-router-af)# neighbor 10.0.0.2 route-reflector-client`	"neighbor 10.0.0.2 default-originate" functionality that automatically gets added to the BGP configuration. The reason to have this is to have the route-reflector get all the VPN prefixes from its peer.
Step 7	**neighbor** {*ip-address* | *peer-group-name*} **send-community extended** **Example:** `Device(config-router-af)# neighbor 10.0.0.2 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor. • The *ip-address* argument specifies the IP address of the BGP-speaking neighbor. • The *peer-group-name* argument specifies the name of a BGP peer group.
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 rtfilter all** **Example:** `Device# show ip bgp rtfilter all`	(Optional) Displays all BGP RT filter information.
Step 10	**show ip bgp rtfilter all summary** **Example:** `Device# show ip bgp rtfilter all summary`	(Optional) Displays summary BGP RT filter information.
Step 11	**show ip bgp vpnv4 all** **Example:** `Device# show ip bgp vpnv4 all`	(Optional) Displays summary BGP VPNv4 information.

Configuration Examples for BGP: RT Constrained Route Distribution

Example: BGP RT Constrained Route Distribution Between a PE and RR

The following example provides the configurations of the routers in the figure below. PE1 and PE2 are each connected to the RR and belong to AS 65000.

Figure 57: BGP: RT Constrained Route Distribution Between a PE and RR



PE1 Configuration

```
ip vrf BLUE
 rd 3:3
 route-target export 1:100
 route-target import 1:100
!
router bgp 65000
 bgp log-neighbor-changes
 neighbor 192.168.2.2 remote-as 65000
 neighbor 192.168.2.2 update-source Loopback0
 no auto-summary
 !
 address-family vpnv4
  neighbor 192.168.2.2 activate
  neighbor 192.168.2.2 send-community extended
 exit-address-family
 !
 address-family rtfilter unicast
  neighbor 192.168.2.2 activate
  neighbor 192.168.2.2 send-community extended
 exit-address-family
 !
 address-family ipv4 vrf BLUE
  redistribute static
 exit-address-family
 !
ip route vrf BLUE 51.51.51.51 255.255.255.255 Null0
!
```

RR Configuration

```
!
router bgp 65000
 bgp log-neighbor-changes
 neighbor 192.168.6.6 remote-as 65000
 neighbor 192.168.6.6 update-source Loopback0
 neighbor 192.168.7.7 remote-as 65000
 neighbor 192.168.7.7 update-source Loopback0
 !
 address-family vpnv4
  neighbor 192.168.6.6 activate
  neighbor 192.168.6.6 send-community extended
  neighbor 192.168.6.6 route-reflector-client
  neighbor 192.168.7.7 activate
  neighbor 192.168.7.7 send-community extended
  neighbor 192.168.7.7 route-reflector-client
```

```
 exit-address-family
 !
 address-family rtfilter unicast
  neighbor 192.168.6.6 activate
  neighbor 192.168.6.6 send-community extended
  neighbor 192.168.6.6 route-reflector-client
  neighbor 192.168.6.6 default-originate
  neighbor 192.168.7.7 activate
  neighbor 192.168.7.7 send-community extended
  neighbor 192.168.7.7 route-reflector-client
  neighbor 192.168.7.7 default-originate
 exit-address-family
!
```

PE2 Configuration

```
!
ip vrf RED
 rd 17:17
 route-target export 150:15
 route-target import 150:1
 route-target import 1:100
!
router bgp 65000
 bgp log-neighbor-changes
 neighbor 192.168.2.2 remote-as 65000
 neighbor 192.168.2.2 update-source Loopback0
 neighbor 192.168.2.2 weight 333
 no auto-summary
 !
 address-family vpnv4
  neighbor 192.168.2.2 activate
  neighbor 192.168.2.2 send-community extended
exit-address-family
 !
 address-family rtfilter unicast
  neighbor 192.168.2.2 activate
  neighbor 192.168.2.2 send-community extended
  exit-address-family
!
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
L3VPNs and route targets	"Configuring MPLS Layer 3 VPNs" module in the *MPLS: Layer 3 VPNs Configuration Guide*
MPLS commands	Cisco IOS Multiprotocol Label Switching Command Reference

MIBs

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

RFCs

RFC	Title
RFC 4360	*BGP Extended Communities Attribute*
RFC 4684	*Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*
RFC 5291	*Outbound Route Filtering 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: RT Constrained Route Distribution

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 51: Feature Information for BGP: RT Constrained Route Distribution

Feature Name	Releases	Feature Information
BGP: RT Constrained Route Distribution	15.1(1)S 15.2(3)T 15.2(4)S 15.1(1)SY	BGP: Route Target (RT) Constrained Route Distribution is a feature that service providers can use in MPLS L3VPNs to reduce the number of unnecessary routes that RRs send to PEs, and thereby save resources. The following commands were introduced: **address-family rtfilter unicast** and **show ip bgp rtfilter** . In Cisco IOS Release 15.2(4)S, support was added for the Cisco 7200 series router.

CHAPTER 39

BGP Best External

The BGP Best External feature provides the network with a backup external route to avoid loss of connectivity of the primary external route. The BGP Best External feature advertises the most preferred route among those received from external neighbors as a backup route. This feature is beneficial in active-backup topologies, where service providers use routing policies that cause a border router to choose a path received over an Interior Border Gateway Protocol (iBGP) session (of another border router) as the best path for a prefix even if it has an Exterior Border Gateway Protocol (eBGP) learned path. This active-backup topology defines one exit or egress point for the prefix in the autonomous system and uses the other points as backups if the primary link or eBGP peering is unavailable. The policy causes the border router to hide the paths learned over its eBGP sessions from the autonomous system because it does not advertise any path for such prefixes. To cope with this situation, some devices advertise one externally learned path called the best external path.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information for BGP Best External, on page 696.

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

Prerequisites for BGP Best External

- The Bidirectional Forwarding Detection (BFD) protocol must be enabled to quickly detect link failures.

- Ensure that the BGP and the Multiprotocol Label Switching (MPLS) network is up and running with the customer site connected to the provider site by more than one path (multihomed).
- The backup path must have a unique next hop that is not the same as the next hop of the best path.
- BGP must support lossless switchover between operational paths.

Restrictions for BGP Best External

- The BGP Best External feature will not install a backup path if BGP Multipath is installed and a multipath exists in the BGP table. One of the multipaths automatically acts as a backup for the other paths.
- The BGP Best External feature is not supported with the following features:
  - MPLS VPN Carrier Supporting Carrier
  - MPLS VPN Inter Autonomous Systems, option B
  - MPLS VPN Virtual Routing and Forwarding Label
- The BGP Best External feature cannot be configured with Multicast or Layer 2 VPN (L2VPN) Virtual Routing and Forwarding address families.
- The BGP Best External feature cannot be configured on route reflectors.
- The BGP Best External feature does not support Nonstop Forwarding/Stateful Switchover (NSF/SSO). However, In-Service Software Upgrade (ISSU) is supported if both Route Processors have the BGP Best External feature configured.
- The BGP Best External feature can be configured only on VPNv4, VPNv6, IPv4 VRF, and IPv6 VRF address families.
- When you configure the BGP Best External feature using the **bgp advertise-best-external** command, you need not enable the BGP Prefix-Independent Convergence (PIC) feature with the **bgp additional-paths install** command. The BGP PIC feature is automatically enabled by the BGP Best External feature.
- When you configure the BGP Best External feature, it will override the functionality of the MPLS VPN--BGP Local Convergence feature. However, you need not remove the **protection local-prefixes** command from the configuration.

Information About BGP Best External

BGP Best External Overview

Service providers use routing policies that cause a border router to choose a path received over an internal BGP (iBGP) session (of another border router) as the best path for a prefix even if it has an external BGP (eBGP) learned path. This practice is popularly known as active-backup topology and is done to define one exit or egress point for the prefix in the autonomous system and to use the other points as backups if the primary link or eBGP peering is unavailable.

The policy, though beneficial, causes the border router to hide the paths learned over its eBGP sessions from the autonomous system because the border router does not advertise any path for such prefixes. To cope with this situation, some devices advertise one externally learned path called the best external path. The best external behavior causes the BGP selection process to select two paths to every destination:

- The best path is selected from the complete set of routes known to that destination.
- The best external path is selected from the set of routes received from its external peers.

BGP advertises the best path to external peers. Instead of withdrawing the best path from its internal peers when it selects an iBGP path as the best path, BGP advertises the best external path to the internal peers.

The BGP Best External feature is an essential component of the Prefix-Independent Convergence (PIC) edge for both Internet access and Multiprotocol Label Switching (MPLS) VPN scenarios and makes alternate paths available in the network in the active-backup topology.

What the Best External Route Means

The BGP Best External feature uses a “best external route” as a backup path, which, according to draft-marques-idr-best-external, is the most preferred route among those received from external neighbors. The most preferred route from external neighbors can be the following:

- Two routers in different clusters that have an Interior Border Gateway Protocol (iBGP) session between them.
- Two routers in different autonomous systems of a confederation that have an External Border Gateway Protocol (eBGP) session between them.

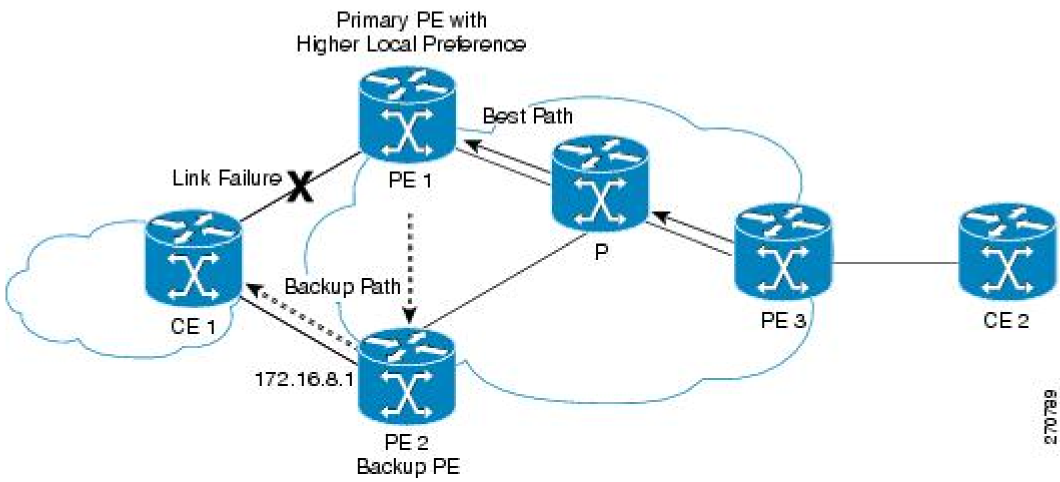
The best external route might be different from the best route installed in the Routing Information Base (RIB). The best route could be an internal route. By allowing the best external route to be advertised and stored, in addition to the best route, networks gain faster restoration of connectivity by providing additional paths that may be used if the primary path fails.

BGP Best External Feature Operation

The BGP Best External feature is based on Internet Engineering Task Force (IETF) draft-marques-idr-best-external.txt. The BGP Best External feature advertises a best external route to its internal peers as a backup route. The backup route is stored in the RIB and Cisco Express Forwarding. If the primary path fails, the Border Gateway Protocol (BGP) Prefix-Independent Convergence (PIC) functionality enables the best external path to take over, enabling faster restoration of connectivity.

Figure 58: MPLS VPN: Best External at the Edge of MPLS VPN, on page 688 shows a Multiprotocol Label Switching (MPLS) VPN using the BGP Best External feature. The network includes the following components:

Figure 58: MPLS VPN: Best External at the Edge of MPLS VPN



- Exterior BGP (eBGP) sessions exist between the provider edge (PE) and customer edge (CE) routers.
- PE 1 is the primary router and has a higher local preference setting.
- Traffic from CE 2 uses PE 1 to reach router CE 1.
- PE 1 has two paths to reach CE 1.
- CE 1 is dual-homed with PE 1 and PE 2.
- PE 1 is the primary path and PE 2 is the backup path.

In Figure 58: MPLS VPN: Best External at the Edge of MPLS VPN, on page 688, traffic in the MPLS cloud flows through PE 1 to reach CE 1. Therefore, PE 2 uses PE 1 as the best path and PE 2 as the backup path.

PE 1 and PE 2 are configured with the BGP Best External feature. BGP computes both the best path (the PE 1-CE 1 link) and a backup path (PE 2) and installs both paths into the Routing Information Base (RIB) and Cisco Express Forwarding. The best external path (PE 2) is advertised to the peer routers, in addition to the best path.

When Cisco Express Forwarding detects a link failure on the PE 1-CE 1 link, Cisco Express Forwarding immediately switches to the backup path PE 2. Traffic is quickly rerouted due to local fast convergence in Cisco Express Forwarding using the backup path. Thus, traffic loss is minimized and fast convergence is achieved.

Configuration Modes for Enabling BGP Best External

You can enable the BGP Best External feature in different modes, each of which protects Virtual Routing and Forwarding (VRF) in its own way:

- If you issue the **bgp advertise-best-external** command in VPNv4 address family configuration mode, it applies to all IPv4 VRFs. If you issue the command in this mode, you need not issue it for specific VRFs.
- If you issue the **bgp advertise-best-external** command in IPv4 address family configuration mode, it applies only to that VRF.

How to Configure BGP Best External

Configuring the BGP Best External Feature

Perform the following task to configure the BGP Best External feature. This task shows how to configure the BGP Best External feature in either an IPv4 or VPNv4 address family. In VPNv4 address family configuration mode, the BGP Best External feature applies to all IPv4 Virtual Routing Forwarding (VRF); you need not configure it for specific VRFs. If you issue the **bgp advertise-best-external** command in IPv4 VRF address family configuration mode, the BGP Best External feature applies only to that VRF.

Before you begin

- Configure the MPLS VPN and verify that it is working properly before configuring the BGP Best External feature. See the "Configuring MPLS Layer 3 VPNs" section for more information.
- Configure multiprotocol VRFs to allow you to share route-target policies (import and export) between IPv4 and IPv6 or configure separate route-target policies for IPv4 and IPv6 VPNs. For information about configuring multiprotocol VRFs, see the "MPLS VPN--VRF CLI for IPv4 and IPv6 VPNs section".
- Ensure that the customer edge (CE) router is connected to the network by at least two paths.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. Do one of the following:
   - **address-family ipv4** [**unicast** | **vrf** *vrf-name*]
   - or
   - **address-family vpnv4** [**unicast**]
   - or
5. **bgp advertise-best-external**
6. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
7. **neighbor** *ip-address* **activate**
8. **neighbor** *ip-address* **fall-over** [**bfd** | **route-map** *map-name*]
9. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.

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

Step 5	**bgp advertise-best-external** **Example:** `Router(config-router-af)# bgp advertise-best-external`	Calculates and uses an external backup path and installs it into the RIB and Cisco Express Forwarding.
Step 6	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Router(config-router-af)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types.
Step 7	**neighbor** *ip-address* **activate** **Example:** `Router(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local router.

	Command or Action	Purpose
Step 8	**neighbor** *ip-address* **fall-over** [**bfd** | **route-map** *map-name*] **Example:** `Router(config-router-af)# neighbor 192.168.1.1 fall-over bfd`	Configures the BGP peering to use fast session deactivation and enables BFD protocol support for failover. • BGP will remove all routes learned through this peer if the session is deactivated.
Step 9	**end** **Example:** `Router(config-router-af)# end`	(Optional) Exits address family configuration mode and returns to privileged EXEC mode.

Verifying the BGP Best External Feature

Perform the following task to verify that the BGP Best External feature is configured correctly.

SUMMARY STEPS

1. **enable**
2. **show vrf detail**
3. • **show ip bgp ipv4** {**mdt** {**all** |**rd** | **vrf**} | **multicast** | **tunnel** | **unicast**}
   • or
   • **show ip bgp vpnv4**{**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} [**rib-failure**] [*ip-prefix/length* [**longer-prefixes**]] [*network-address* [*mask*] [**longer-prefixes**]] [**cidr-only**] [**community**] [*community-list*] [**dampened-paths**] [**filter-list**] [**flap-statistics**] [**inconsistent-as**] [**neighbors**] [**paths** [**line**]] [**peer-group**] [**quote-regexp**] [**regexp**] [**summary**] [**labels**]
4. **show bgp vpnv4 unicast vrf** *vrf-name ip-address*
5. **show ip route vrf** *vrf-name* **repair-paths** *ip-address*
6. **show ip cef vrf** *vrf-name ip-address* **detail**

DETAILED STEPS

Step 1 **enable**

Use this command to enable privileged EXEC mode. Enter your password, if prompted. For example:

Example:

```
Router> enable
Router#
```

Step 2 **show vrf detail**

Use this command to verify that the BGP Best External feature is enabled. The following **show vrf detail** command output shows that the BGP Best External feature is enabled.

Example:

```
Router# show vrf detail

VRF test1 (VRF Id = 1); default RD 400:1; default VPNID <not set>
  Interfaces:
    Se4/0
Address family ipv4 (Table ID = 1 (0x1)):
  Export VPN route-target communities
    RT:100:1                 RT:200:1                 RT:300:1
    RT:400:1
  Import VPN route-target communities
    RT:100:1                 RT:200:1                 RT:300:1
    RT:400:1
  No import route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix

 Prefix protection with additional path enabled
Address family ipv6 not active.
```

Step 3

- **show ip bgp ipv4** {**mdt** {**all** |**rd** | **vrf**} | **multicast** | **tunnel** | **unicast**}
- or
- **show ip bgp vpnv4**{**all** | **rd** *route-distinguisher* | **vrf** *vrf-name*} [**rib-failure**] [*ip-prefix/length* [**longer-prefixes**]] [*network-address* [*mask*] [**longer-prefixes**]] [**cidr-only**] [**community**] [*community-list*] [**dampened-paths**] [**filter-list**] [**flap-statistics**] [**inconsistent-as**] [**neighbors**] [**paths** [**line**]] [**peer-group**] [**quote-regexp**] [**regexp**] [**summary**] [**labels**]

Use this command to verify that the best external route is advertised. In the command output, the code b indicates a backup path and the code x designates the best external path.

Example:

```
Router# show ip bgp vpnv4 all

BGP table version is 1104964, local router ID is 10.2.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, multipath,
b backup-path, x best-external
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 11:12 (default for vrf blue)
*>i1.0.0.1/32       10.10.3.3                0    200      0 1 ?
* i                 10.10.3.3                0    200      0 1 ?
*                   10.0.0.1                               0 1 ?
*bx                 10.0.0.1                 0             0 1 ?
*                   10.0.0.1                               0 1 ?
```

Step 4 **show bgp vpnv4 unicast vrf** *vrf-name ip-address*

Use this command to verify that the best external route is advertised.

Example:

```
Router# show bgp vpnv4 unicast vrf vpn1 10.10.10.10
BGP routing table entry for 10:10:10.10.10.10/32, version 10
Paths: (2 available, best #1, table vpn1)
  Advertise-best-external
  Advertised to update-groups:
    1          2
  200
    10.6.6.6 (metric 21) from 10.6.6.6 (10.6.6.6)
```

```
    Origin incomplete, metric 0, localpref 200, valid, internal, best
    Extended Community: RT:1:1
    mpls labels in/out 23/23
  200
    10.1.2.1 from 10.1.2.1 (10.1.1.1)
    Origin incomplete, metric 0, localpref 100, valid,
external, backup/repair,     advertise-best-external
    Extended Community: RT:1:1 , recursive-via-connected
    mpls labels in/out 23/nolabel
```

Step 5 **show ip route vrf** *vrf-name* **repair-paths** *ip-address*

Use this command to display the repair route.

Example:

```
Router# show ip route vrf vpn1 repair-paths

Routing Table: vpn1
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP
       + - replicated route, % - next hop override
Gateway of last resort is not set
      10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
B        10.1.1.0/24 [200/0] via 10.6.6.6, 00:38:33
                    [RPR][200/0] via 10.1.2.1, 00:38:33
B        10.1.1.1/32 [200/0] via 10.6.6.6, 00:38:33
                    [RPR][200/0] via 10.1.2.1, 00:38:33
      10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
C        10.1.2.0/24 is directly connected, Ethernet0/0
L        10.1.2.2/32 is directly connected, Ethernet0/0
B        10.1.6.0/24 [200/0] via 10.6.6.6, 00:38:33
                     [RPR][200/0] via 10.1.2.1, 00:38:33
```

Step 6 **show ip cef vrf** *vrf-name ip-address* **detail**

Use this command to display the best external route.

Example:

```
Router# show ip cef vrf test 10.71.8.164 detail

10.71.8.164/30, epoch 0, flags rib defined all labels
 recursive via 10.249.0.102 label 35
   nexthop 10.249.246.101 Ethernet0/0 label 25
 recursive via 10.249.0.104 label 28,
repair
   nexthop 10.249.246.101 Ethernet0/0 label 24
```

Configuration Examples for BGP Best External

Example: Configuring the BGP Best External Feature

The following example shows how to configure the BGP Best External feature in VPNv4 mode:

```
vrf definition test1
 rd 400:1
 route-target export 100:1
 route-target export 200:1
 route-target export 300:1
 route-target export 400:1
 route-target import 100:1
 route-target import 200:1
 route-target import 300:1
 route-target import 400:1
 address-family ipv4
 exit-address-family
 exit
!
interface Ethernet1/0
 vrf forwarding test1
 ip address 10.0.0.1 255.0.0.0
 exit
!
router bgp 64500
 no synchronization
 bgp log-neighbor-changes
 neighbor 10.5.5.5 remote-as 64500
 neighbor 10.5.5.5 update-source Loopback0
 neighbor 10.6.6.6 remote-as 64500
 neighbor 10.6.6.6 update-source Loopback0
 no auto-summary
 !
 address-family vpnv4

bgp advertise-best-external
  neighbor 10.5.5.5 activate
  neighbor 10.5.5.5 send-community extended
  neighbor 10.6.6.6 activate
  neighbor 10.6.6.6 send-community extended
 exit-address-family
 !
 address-family ipv4 vrf test1
  no synchronization
bgp recursion host
  neighbor 192.168.13.2 remote-as 64511
  neighbor 192.168.13.2 fall-over bfd
  neighbor 192.168.13.2 activate
  neighbor 192.168.13.2 as-override
 exit-address-family
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
Basic MPLS VPNs	"Configuring MPLS Layer 3 VPNs" module in the *MPLS: Layer 3 VPNs Configuration Guide*
Multiprotocol VRFs	"MPLS VPN VRF CLI for IPv4 and IPv6 VPNs" module in the *MPLS: Layer 3 VPNs Configuration Guide*
A failover feature that creates a new path after a link or node failure	MPLS VPN--BGP Local Convergence

Standards

Standard	Title
draft-marques-idr-best-external	*BGP Best External, Advertisement of the best external route to iBGP*

MIBs

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

RFCs

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

Technical Assistance

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

Feature Information for BGP Best External

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 52: Feature Information for BGP Best External

Feature Name	Releases	Feature Information
BGP Best External	12.2(33)SRE 15.2(3)T 15.2(4)S 15.1(1)SY	The BGP Best External feature provides the network with a backup external route to avoid loss of connectivity of the primary external route. This feature advertises the most preferred route among those received from external neighbors as a backup route. The following commands were introduced or modified: **bgp advertise-best-external**, **bgp recursion host**, **show ip bgp**, **show ip bgp vpnv4**, **show ip cef**, **show ip cef vrf**, **show ip route**, and **show ip route vrf**.

CHAPTER 40

BGP IPv6 PIC Edge and Core for IP/MPLS

The BGP IPv6 PIC Edge and Core for IP/MPLS feature improves convergence for both core and edge failures after a network failure.

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 IPv6 IPC Edge and Core for IP/MPLS

BGP IPv6 PIC Edge for IP MPLS

The BGP IPv6 PIC Edge for IP MPLS feature improves convergence for both core and edge failures after a network failure. The BGP IPv6 PIC edge for IP MPLS feature creates and stores a backup or alternate path in the Routing Information Base (RIB), the Forwarding Information Base (FIB), and in Cisco Express Forwarding, so that the backup or alternate path can immediately take over wherever a failure is detected, thus enabling fast failover.

How to Configure BGP IPv6 PIC Edge and Core for IP/MPLS

Configuring BGP IPv6 PIC Edge for IP MPLS

Because many service provider networks contain many VRFs, the BGP PIC feature allows you to configure BGP PIC feature for all VRFs at once. Performing this task in IPv6 address family configuration mode protects IPv6 VRFs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**]
5. **bgp additional-paths install**
6. **bgp recursion host**

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 100`	Configures the BGP routing process.
Step 4	**address-family ipv6** [**vrf** *vrf-name*] [**unicast** | **multicast** | **vpnv6**] **Example:** `Device(config-router)# address-family ipv6 vrf_pic`	Specifies a VRF table named vrf_pic, and enters IPv6 address family configuration mode.
Step 5	**bgp additional-paths install** **Example:**	Calculates a backup path and installs it into the RIB and Cisco Express Forwarding.

	Command or Action	Purpose
	`Device(config-router-af)# bgp additional-paths install`	
Step 6	**bgp recursion host** **Example:** `Device(config-router-af)# bgp recursion host`	Enables the recursive-via-host flag for IPv6 address families.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
BGP PIC Edge for IP and MPLS-VPN	"BGP PIC Edge for IP and MPLS-VPN" module in the *IP Routing: BGP Configuration Guide*

Technical Assistance

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

Feature Information for BGP IPv6 PIC Edge and Core for IP/MPLS

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 53: Feature Information for BGP IPv6 PIC Edge and Core for IP/MPLS

Feature Name	Releases	Feature Information
BGP IPv6 PIC Edge and Core for IP/MPLS	Cisco IOS XE Release 3.3S 15.2(3)T 15.2(4)S	The BGP IPv6 PIC Edge and Core for IP/MPLS feature improves convergence for both core and edge failures after a network failure. The following commands were modified: **bgp additional-paths install**, **bgp advertise-best-external**, **bgp recursion host**. In Cisco IOS Release 15.2(4)S, support was added for the Cisco 7200 series router.

CHAPTER 41

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

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

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

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information 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 http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Prerequisites for BGP PIC

- Ensure that the Border Gateway Protocol (BGP) and the IP or Multiprotocol Label Switching (MPLS) network is up and running with the customer site connected to the provider site by more than one path (multihomed).
- Ensure that the backup/alternate path has a unique next hop that is not the same as the next hop of the best path.
- Enable the Bidirectional Forwarding Detection (BFD) protocol to quickly detect link failures of directly connected neighbors.
- Ensure that 2KB memory is available per prefix on the RP, SP and each line card. For example, if you need to scale up to 100000 prefixes, ensure that at least 200 MB is free on the RP, SP and each line card.

Restrictions for BGP PIC

- With BGP Multipath, the BGP Prefix-Independent Convergence (PIC) feature is already supported.
- In MPLS VPNs, the BGP PIC feature is not supported with MPLS VPN Inter-Autonomous Systems Option B.
- The BGP PIC feature supports prefixes only for IPv4, IPv6, VPNv4, and VPNv6 address families.
- The BGP PIC feature cannot be configured with Multicast or L2VPN Virtual Routing and Forwarding (VRF) address families.
- If the route reflector is only in the control plane, then you do not need BGP PIC, because BGP PIC addresses data plane convergence.
- When two PE routers become each other's backup/alternate path to a CE router, traffic might loop if the CE router fails. Neither router will reach the CE router, and traffic will continue to be forwarded between the PE routers until the time-to-live (TTL) timer expires.
- The BGP PIC feature does not support Nonstop Forwarding with Stateful Switchover (NSF/SSO). However, ISSU is supported if both Route Processors have the BGP PIC feature configured.
- The BGP PIC feature solves the traffic forwarding only for a single network failure at both the edge and the core.
- The BGP PIC feature does not work with the BGP Best External feature. If you try to configure the BGP PIC feature after configuring the BGP Best External feature, you receive an error.

About BGP PIC

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

Benefits

- An extra path for failover allows faster restoration of connectivity when a primary path is invalid or withdrawn.
- Reduction of traffic loss.
- Constant convergence time so that the switching time is the same for all prefixes.

BGP Convergence

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

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

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

Improve Convergence

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

- BGP Functionality

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

- RIB Functionality

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

- Cisco Express Forwarding Functionality

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

- MPLS Functionality

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

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

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

Convergence in the Data Plane

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

Convergence in the Control Plane

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

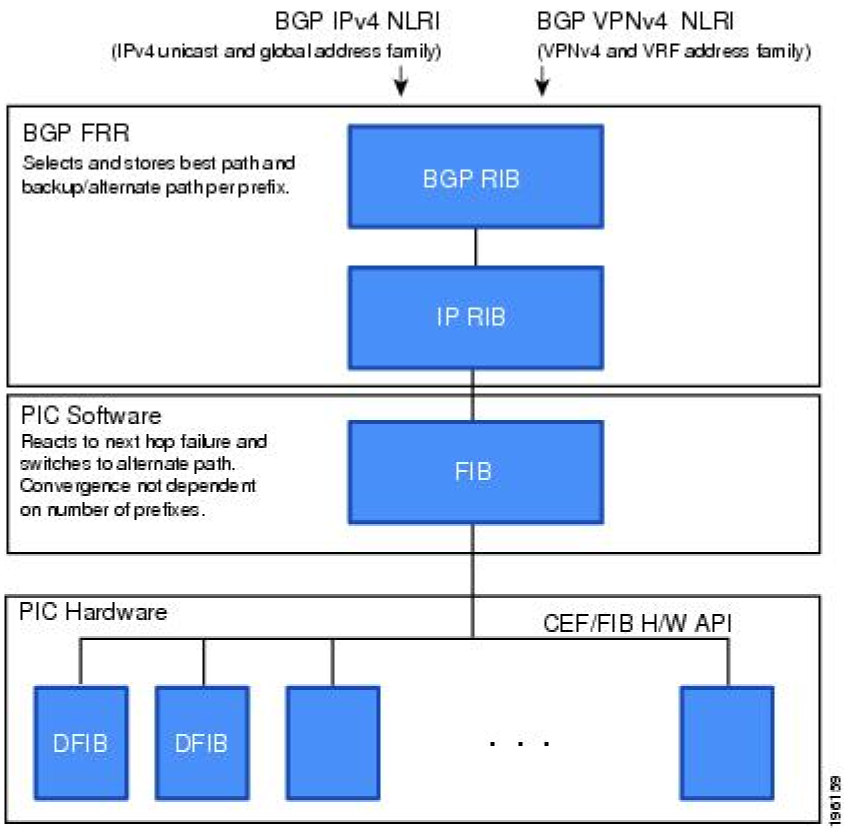
BGP Fast Reroute

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

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

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

Figure 59: BGP PIC Edge and BGP FRR



Detect a Failure

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

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

How BGP PIC Achieves Subsecond Convergence

The BGP PIC feature works at the Cisco Express Forwarding level, and Cisco Express Forwarding can be processed in both hardware line cards and in the software.

- For platforms that support Cisco Express Forwarding processing in the line cards, the BGP PIC feature can converge in subseconds. The Cisco 7600 router and Cisco 10000 router supports Cisco Express Forwarding processing in the line cards and in the software, and thus can attain subsecond convergence.
- For platforms that do not use Cisco Express Forwarding in hardware line cards, Cisco Express Forwarding is achieved in the software. The BGP PIC feature will work with the Cisco Express Forwarding through the software and achieve convergence within seconds. The Cisco 7200 router supports Cisco Express Forwarding in the software and thus can achieve convergence in seconds rather than milliseconds.

MPLS VPN–BGP Local Convergence

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

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

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

Enable BGP PIC

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

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

BGP PIC Scenario

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

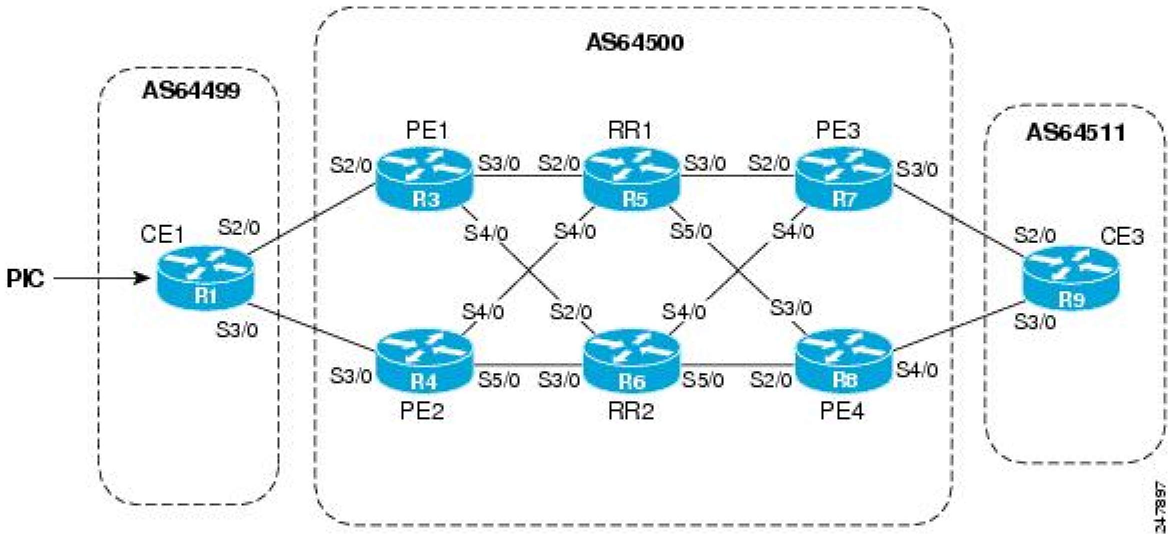
IP PE-CE Link and Node Protection

The network includes the following components:

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

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

Figure 60: Using BGP PIC to Protect the PE-CE Link



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

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

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

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

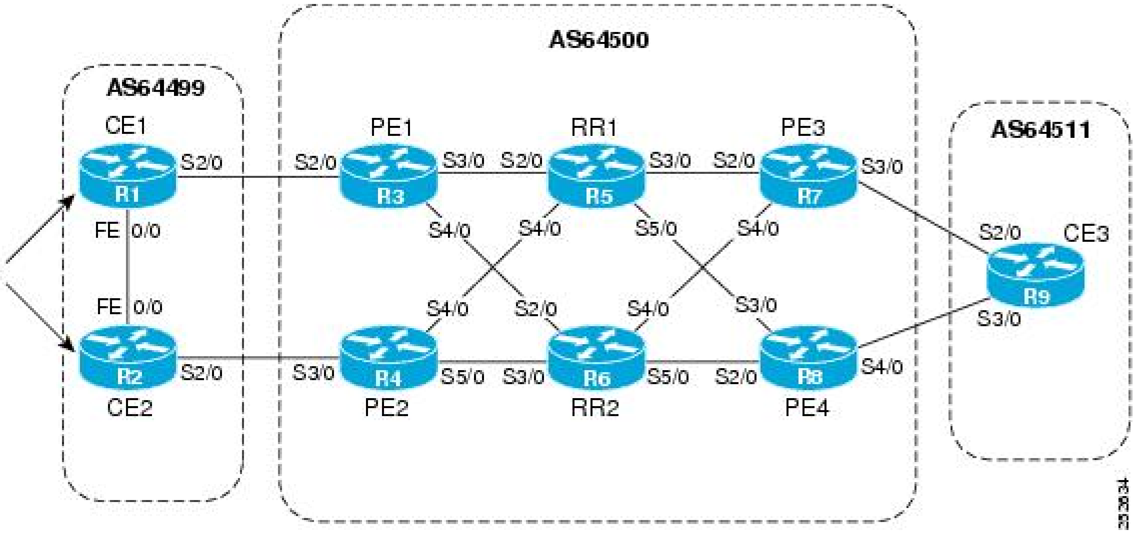
There should not be any policies set on CE1 and CE2 for the eBGP peers PE1 and PE2. Both CE routers must point to the eBGP route as next hop. On CE1, the next hop to reach CE3 is through PE1, so PE1 is the best path to reach CE3. On CE2, the best path to reach CE3 is PE2. CE2 advertises itself as the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

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

If the CE1-PE1 link or PE1 goes down and BGP PIC is enabled on CE1, BGP recomputes the best path, removing the next hop PE1 from RIB and reinstalling CE2 as the next hop into the RIB and Cisco Express

Forwarding. CE1 automatically gets a backup/alternate repair path into Cisco Express Forwarding and the traffic loss during forwarding is now in subseconds, thereby achieving fast convergence.

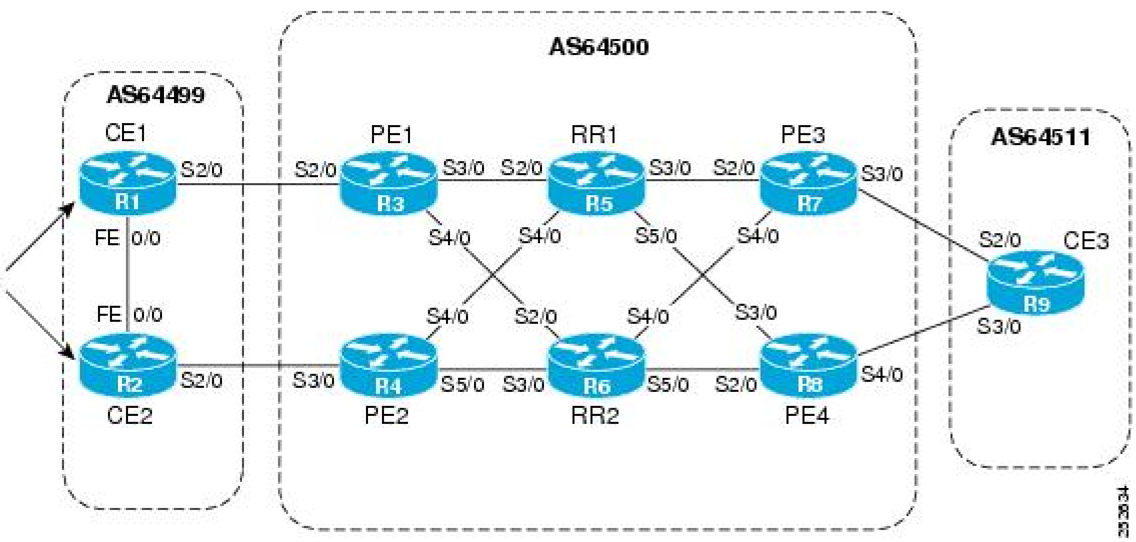
Figure 61: Using BGP PIC in a Dual CE, Dual PE Network



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

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

Figure 62: Using BGP PIC in a Dual CE, Dual PE Network



- eBGP sessions exist between the PE and CE routers.
- The PE routers are VPNv4 iBGP peers with reflect routers in the MPLS network.
- Traffic from CE1 uses PE1 to reach the network 192.168.9.0/24 through router CE3.
- CE3 is dual-homed with PE3 and PE4.

- PE1 has two paths to reach CE3 from the reflect routers:
    - PE3 is the primary path with the next hop as a PE3 address.
    - PE4 is the backup/alternate path with the next hop as a PE4 address.

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

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

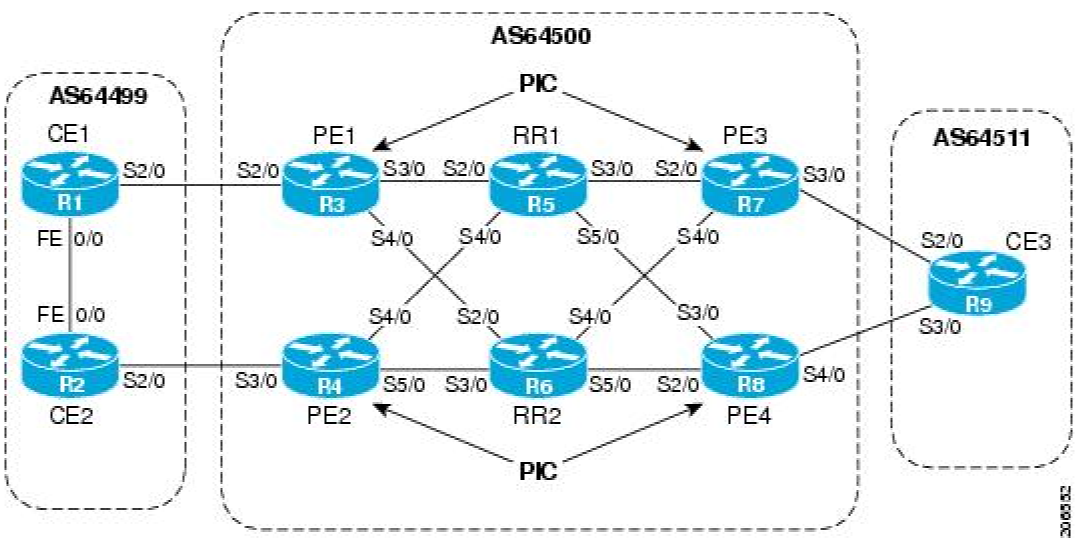
When the PE3-CE3 link goes down, Cisco Express Forwarding detects the link failure, and PE3 recomputes the best path, selects PE4 as the best path, and sends a withdraw message for the PE3 prefix to the reflect routers. Some of the traffic goes through PE3-PE4 until BGP installs PE4 as the best path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

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

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

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

Figure 63: Enabling BGP PIC on All PEs Routers in the MPLS Network



The network includes the following components:

- eBGP sessions exist between the PE and CE routers.
- The PE routers are VPNv4 iBGP peers with reflect routers in the MPLS network.
- Traffic from CE1 uses PE1 to reach the network 192.168.9.0/24 through router CE3.

- CE3 is dual-homed with PE3 and PE4.
- PE1 has two paths to reach CE3 from the reflect routers:
  - PE3 is the primary path with the next hop as a PE3 address.
  - PE4 is the backup/alternate path with the next hop as a PE4 address.

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

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

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

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

No Local Policies Set on the PE Routers

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

Local Policies Set on the PE Routers

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

Cisco Express Forwarding Recursion

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

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

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

- Recursion slows down convergence when Cisco Express Forwarding searches all the FIB entries.

- BGP PIC Edge already precomputes an alternate path. It therefore eliminates the need for Cisco Express Forwarding recursion.

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

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

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

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

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

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

How to Configure BGP PIC

Configuring BGP PIC

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

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

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

Before you begin

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

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. Do one of the following:
    - **address-family ipv4** [**unicast** | **vrf** *vrf-name*]
    - or
    - **address-family vpnv4** [**unicast**]
5. **bgp additional-paths install**
6. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
7. **neighbor** *ip-address* **activate**
8. **bgp recursion host**
9. **neighbor** *ip-address* **fall-over** [**bfd** |**route-map** *map-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.

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

	Command or Action	Purpose
Step 5	**bgp additional-paths install** **Example:** `Device(config-router-af)# bgp additional-paths install`	Calculates a backup/alternate path and installs it into the RIB and Cisco Express Forwarding.
Step 6	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 45000`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 multiprotocol BGP neighbor table of the local router. • By default, neighbors that are defined using the **neighbor remote-as** command in router configuration mode exchange only IPv4 unicast address prefixes. To exchange other address prefix types, neighbors must also be activated using the **neighbor activate** command in address family configuration mode for the other prefix types.
Step 7	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 unicast address family with the local router.
Step 8	**bgp recursion host** **Example:** `Device(config-router-af)# bgp recursion host`	(Optional) Enables the recursive-via-host flag for IPv4, VPNv4, and VRF address families. • When the BGP PIC feature is enabled, Cisco Express Forwarding recursion is disabled. Under most circumstances, you do not want to enable recursion when BGP PIC is enabled.
Step 9	**neighbor** *ip-address* **fall-over** [**bfd** |**route-map** *map-name*] **Example:** `Device(config-router-af)# neighbor 192.168.1.1 fall-over bfd`	Enables BFD protocol support to detect when a neighbor has gone away, which can occur within a subsecond.
Step 10	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP PIC

Example: Configuring BGP PIC

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

```
vrf definition test1
 rd 400:1
 route-target export 100:1
 route-target export 200:1
 route-target export 300:1
 route-target export 400:1
 route-target import 100:1
 route-target import 200:1
 route-target import 300:1
 route-target import 400:1
 address-family ipv4
 exit-address-family
 exit
!

 vrf forwarding test1
 ip address 10.0.0.1 255.0.0.0
 exit
router bgp 3
 no synchronization
 bgp log-neighbor-changes
 redistribute static
 redistribute connected
 neighbor 10.6.6.6 remote-as 3
 neighbor 10.6.6.6 update-source Loopback0
 neighbor 10.7.7.7 remote-as 3
 neighbor 10.7.7.7 update-source Loopback0
 no auto-summary
 !
 address-family vpnv4
  bgp additional-paths install
  neighbor 10.6.6.6 activate
  neighbor 10.6.6.6 send-community both
  neighbor 10.7.7.7 activate
  neighbor 10.7.7.7 send-community both
 exit-address-family
 !
 address-family ipv4 vrf blue
  import path selection all
  import path limit 10
  no synchronization
  neighbor 10.11.11.11 remote-as 1
  neighbor 10.11.11.11 activate
 exit-address-family
 !
 address-family ipv4 vrf green
  import path selection all
  import path limit 10
  no synchronization
  neighbor 10.13.13.13 remote-as 1
```

```
  neighbor 10.13.13.13 activate
 exit-address-family
```

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

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

Example: Displaying Backup Alternate Paths for BGP PIC

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

```
Device# show ip bgp vpnv4 vrf blue 10.0.0.0

BGP routing table entry for 10:12:12.0.0.0/24, version 88
Paths: (4 available, best #1, table blue)
  Additional-path
  Advertised to update-groups:
     6
  1, imported path from 12:23:12.0.0.0/24
    10.3.3.3 (metric 21) from 10.6.6.6 (10.6.6.6)
      Origin incomplete, metric 0, localpref 200, valid, internal, best
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
  1, imported path from 12:23:12.0.0.0/24
    10.13.13.13 (via green) from 10.13.13.13 (10.0.0.2)
      Origin incomplete, metric 0, localpref 100, valid, external
      Extended Community: RT:12:23 , recursive-via-connected
  1, imported path from 12:23:12.0.0.0/24
    10.3.3.3 (metric 21) from 10.7.7.7 (10.7.7.7)
      Origin incomplete, metric 0, localpref 200, valid, internal
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
  1
    10.11.11.11 from 10.11.11.11 (1.0.0.1)
      Origin incomplete, metric 0, localpref 100, valid, external, backup/repair
      Extended Community: RT:11:12 , recursive-via-connected
```

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

```
Device# show ip bgp vpnv4 vrf green 12.0.0.0

BGP routing table entry for 12:23:12.0.0.0/24, version 87
```

```
Paths: (4 available, best #4, table green)
  Additional-path
  Advertised to update-groups:
     5
  1, imported path from 11:12:12.0.0.0/24
    10.11.11.11 (via blue) from 10.11.11.11 (1.0.0.1)
      Origin incomplete, metric 0, localpref 100, valid, external
      Extended Community: RT:11:12 , recursive-via-connected
  1
    10.3.3.3 (metric 21) from 10.7.7.7 (10.7.7.7)
      Origin incomplete, metric 0, localpref 200, valid, internal
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
  1
    10.13.13.13 from 10.13.13.13 (10.0.0.2)
      Origin incomplete, metric 0, localpref 100, valid, external, backup/repair
      Extended Community: RT:12:23 , recursive-via-connected
  1
    10.3.3.3 (metric 21) from 10.6.6.6 (10.6.6.6)
      Origin incomplete, metric 0, localpref 200, valid, internal, best
      Extended Community: RT:12:23
      Originator: 10.3.3.3, Cluster list: 10.0.0.1 , recursive-via-host
      mpls labels in/out nolabel/37
```

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

```
Device# show ip bgp 10.0.0.0 255.255.0.0

BGP routing table entry for 10.0.0.0/16, version 123
Paths: (4 available, best #3, table default)
  Additional-path
  Advertised to update-groups:
     2          3
  Local
    10.0.101.4 from 10.0.101.4 (10.3.3.3)
      Origin IGP, localpref 100, weight 500, valid, internal
  Local
    10.0.101.3 from 10.0.101.3 (10.4.4.4)
      Origin IGP, localpref 100, weight 200, valid, internal
  Local
    10.0.101.2 from 10.0.101.2 (10.1.1.1)
      Origin IGP, localpref 100, weight 900, valid, internal, best
  Local
    10.0.101.1 from 10.0.101.1 (10.5.5.5)
      Origin IGP, localpref 100, weight 700, valid, internal, backup/repair
```

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

```
Device# show ip route repair-paths 10.0.0.0 255.255.0.0

Routing entry for 10.0.0.0/16
  Known via "bgp 10", distance 200, metric 0, type internal
  Last update from 10.0.101.2 00:00:56 ago
  Routing Descriptor Blocks:
  * 10.0.101.2, from 10.0.101.2, 00:00:56 ago
      Route metric is 0, traffic share count is 1
      AS Hops 0
      MPLS label: none
    [RPR]10.0.101.1, from 10.0.101.1, 00:00:56 ago
      Route metric is 0, traffic share count is 1
```

```
AS Hops 0
MPLS label: none
```

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

```
Device# show ip cef 10.0.0.0 255.255.0.0 detail

10.0.0.0/16, epoch 0, flags rib only nolabel, rib defined all labels
  recursive via 10.0.101.2
    attached to
  recursive via 10.0.101.1, repair
    attached to
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
Basic MPLS VPNs	Configuring MPLS Layer 3 VPNs
A failover feature that creates a new path after a link or node failure	MPLS VPN--BGP Local Convergence
Configuring multiprotocol VRFs	MPLS VPN--VRF CLI for IPv4 and IPv6 VPNs

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Technical Assistance

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

Feature Information for BGP PIC

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 54: Feature Information for BGP PIC

Feature Name	Releases	Feature Information
BGP PIC Edge for IP and MPLS-VPN	12.2(33)SRE 12.2(33)XNE 15.2(3)T	The BGP PIC Edge for IP and MPLS-VPN feature improves BGP convergence after a network failure. This convergence is applicable to both core and edge failures and can be used in both IP and MPLS networks. The BGP PIC Edge for IP and MPLS-VPN feature creates and stores a backup/alternate path in the routing information base (RIB), forwarding information base (FIB), and Cisco Express Forwarding so that when a failure is detected, the backup/alternate path can immediately take over, thus enabling fast failover. In 12.2(33)SRE, this feature was introduced on the Cisco 7200 and Cisco 7600 routers. In 12.2(33)XNE, support was added for the Cisco 10000 router. The following commands were introduced or modified: **bgp additional-paths install**, **bgp recursion host**, **show ip bgp**, **show ip cef**, **show ip route**, and **show vrf**.

CHAPTER 42

Configuring a BGP Route Server

BGP route server is a feature designed for internet exchange (IX) operators that provides an alternative to full eBGP mesh peering among the service providers who have a presence at the IX. The route server provides eBGP route reflection with customized policy support for each service provider. That is, a route server context can override the normal BGP best path for a prefix with a different path based on a policy, or suppress all paths for a prefix and not advertise the prefix. The BGP route server provides reduced configuration complexity and reduced CPU and memory requirements on each border router. The route server also reduces overhead expense incurred by individualized peering agreements.

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 Server

The Problem That a BGP Route Server Solves

In order to understand the problem that a BGP route server solves, it is helpful to understand the following information about service provider (SP) peering and the eBGP mesh that results from public peering.

Private vs. Public Peering of Service Providers

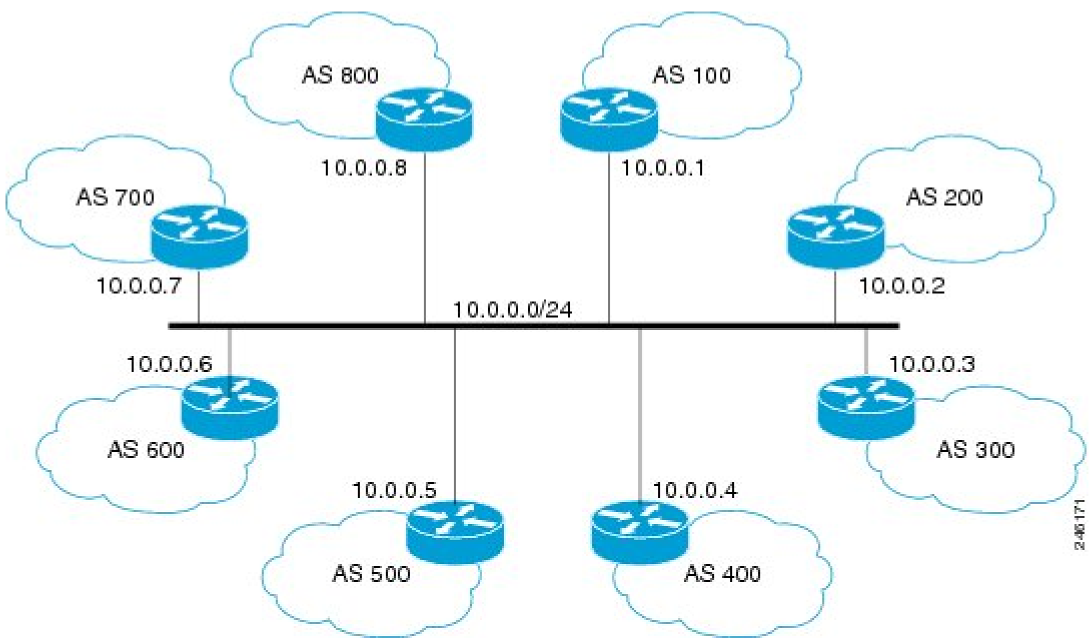
Peering is the connecting of two service providers (SPs) for the purpose of exchanging network traffic between them. Peerings are either private or public.

- In a private peering, two SPs that want to connect decide on a physical site where their networks can be connected and negotiate a contract that covers the details of the connection arrangement. The two parties provide all of the physical space, network equipment, and services (such as electricity and cooling) required to operate the peering connections.
- A public internet exchange (IX), also called a network access point (NAP), is a physical location operated to facilitate the interconnection of multiple SP networks using a shared infrastructure. The IX provides the physical necessities such as rack space for networking devices, electricity, cooling, and a common switching infrastructure required for SPs to directly connect their networks. Unlike private peering, which is typically one-to-one, the IX allows an SP that has a presence at the exchange to connect to multiple peers at a single physical location. The IX provides an alternative to private peering for smaller SPs who do not have the resources required to maintain numerous private peering connections.

Public Peering of SPs within an IX Using BGP

Within the IX, each SP maintains a BGP border router connected to the common switching infrastructure or subnet, as shown in the figure below. In this example, eight different SPs with AS numbers 100 through 800 are connected to the 10.0.0.0/24 subnet through their BGP border routers addressed 10.0.0.1 through 10.0.0.8.

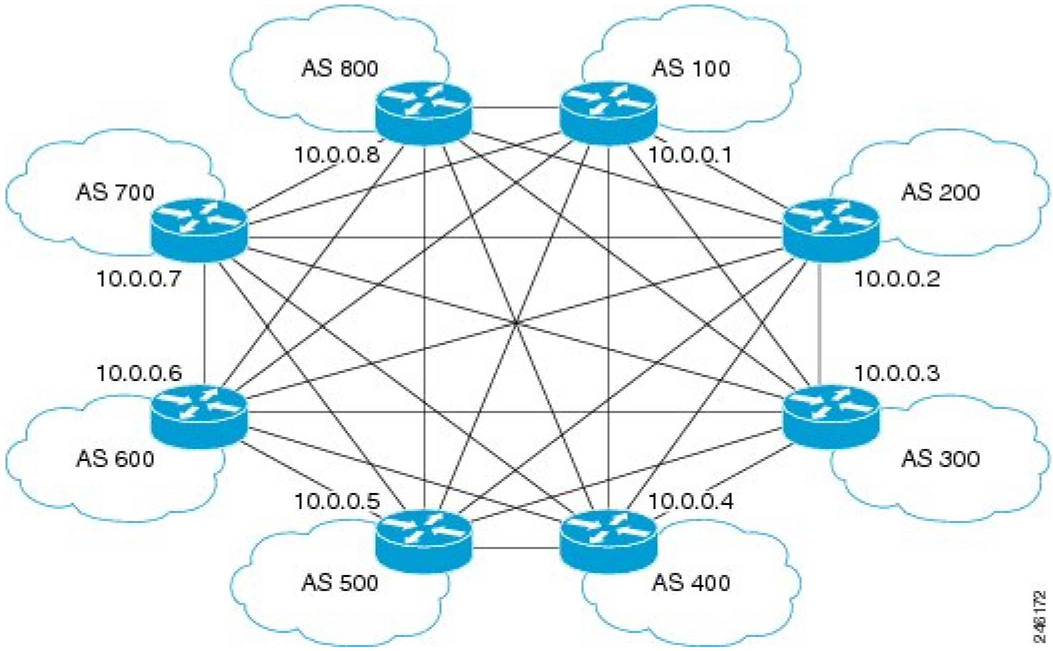
Figure 64: IX Shared Switching Infrastructure



Although each SP's border router is attached to the shared subnet, BGP sessions between each of the SPs must still be configured and maintained individually, for every other SP with which a given SP wants to establish a peering relationship.

Assuming that each SP wants to connect to every other SP, the resulting full mesh of BGP sessions established is shown in the figure below.

Figure 65: IX eBGP Full Mesh



Just as the required iBGP full mesh in an autonomous system presents a scaling and administrative challenge within an SP network, the eBGP full mesh required for peering at an IX presents a challenge for eBGP, for these reasons:

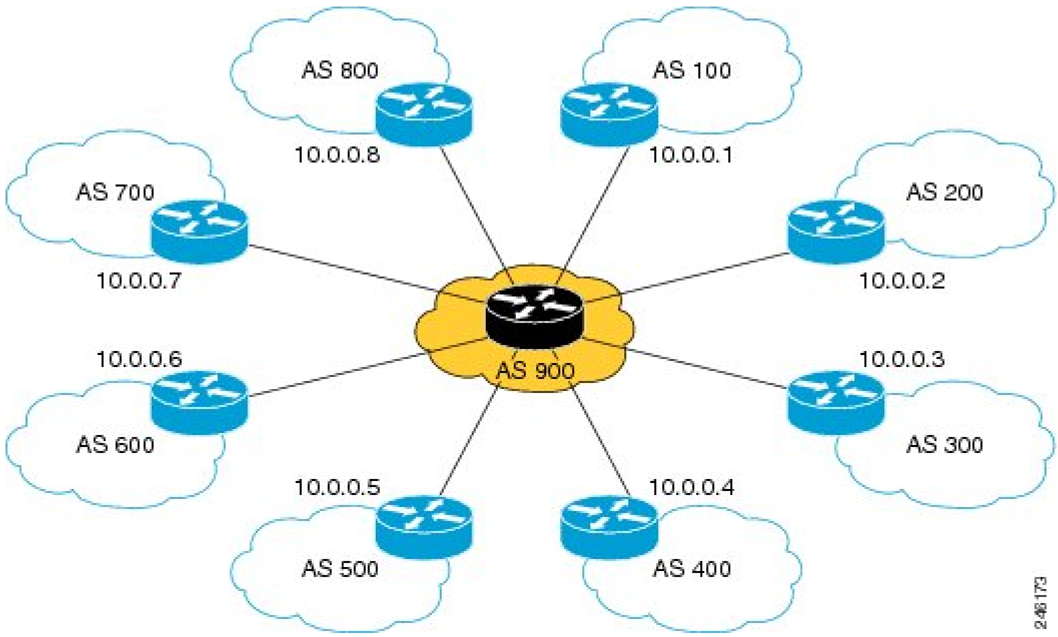
- The full mesh of direct peering sessions requires a BGP session to be configured and maintained for each connection.
- There is additional operational overhead from contracts that would need to be negotiated with each SP peer connecting to a given provider at the IX.

Because larger global SPs might have a presence at dozens or hundreds of internet exchanges worldwide, and dozens or hundreds of potential peers at each IX, it would be a huge operational expense to connect to all of the small providers. Consequently, the state of peering prior to the BGP Route Server feature is that a large global SP connects to only a subset of other large providers to limit the management and operational overhead. A more scalable alternative to direct peering would allow large global SPs to connect to more small providers.

BGP Route Server Simplifies SP Interconnections

A BGP route server simplifies interconnection of SPs at an IX, as shown in the figure below.

Figure 66: IX with eBGP Route Server

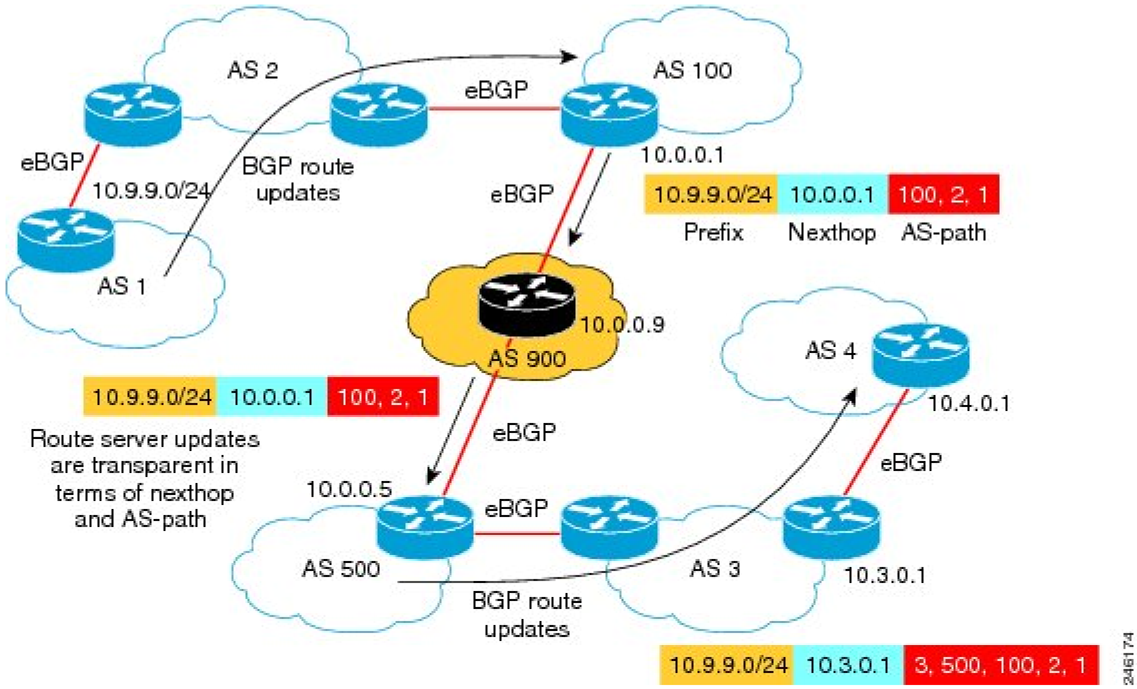


Instead of maintaining individual, direct eBGP peerings with every other provider, an SP maintains only a single connection to the route server operated by the IX. Peering with only the route server reduces the configuration complexity on each border router, reduces CPU and memory requirements on the border routers, and avoids most of the operational overhead incurred by individualized peering agreements.

The route server provides AS-path, MED, and nexthop transparency so that peering SPs at the IX still appear to be directly connected. In reality, the IX route server mediates this peering, but that relationship is invisible outside of the IX.

The figure below illustrates an example of transparent route propagation with a route server at an IX.

Figure 67: Transparent Route Propagation with Route Server at IX



In the figure above, a routing update goes from AS 1 to AS 2 to AS 100. The update leaves the router in AS 100 advertising that the router can reach the prefix 10.9.9.0/24, use 10.0.0.1 as the next hop, and use the AS path of AS100, AS2, AS1.

The router in AS 900 is a route server and the router in AS 500 is a route server client. A route server client receives updates from a route server. As shown in the figure above, the router in AS 900 does not change the update; route server updates are transparent in terms of MED, next hop and AS-path. The update goes to the client with the same prefix, next hop and AS-path that came from the router at 10.0.0.1.

Benefits of a BGP Route Server

A BGP route server provides the following benefits:

- Reduced configuration complexity on each border router.
- Reduced CPU and memory requirements on each border router.
- Reduced operational overhead incurred by individualized peering agreements.
- The ability for a route server context to override the normal BGP best path with an alternative path based on some policy.
- The ability for a route server context to suppress all paths for a prefix and therefore not advertise the prefix.

Route Server Context Provides Flexible Routing Policy

A BGP route server can provide a flexible routing policy. Your network environment might or might not have routes that need a customized (flexible) policy handling.

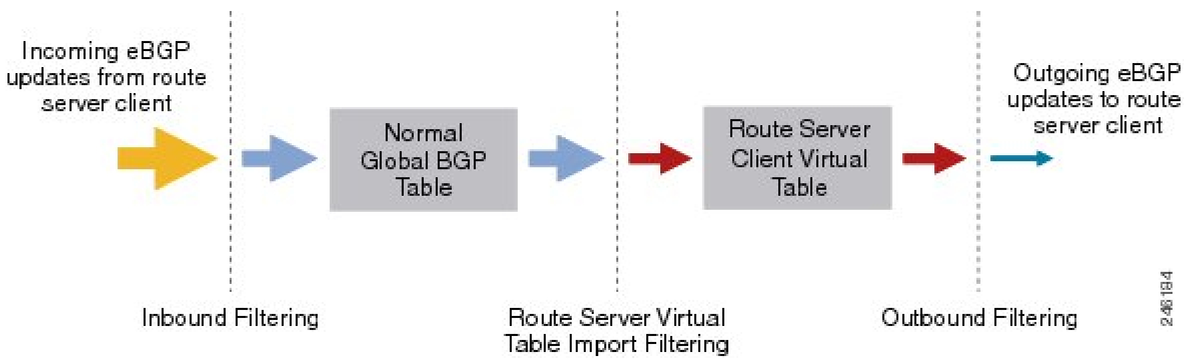
Routes needing flexible policy handling are selected for import into a route server context by configuring an import map. The import map references a route map, where the actual policy is defined by at least one **permit** statement. Routes (paths) that match the route map are included in a second "best path" calculation. The resulting best path, if it is different from the global best path, is imported into the context. Router server clients associated with a router server context will override the global best path with the context's best path when sending routing updates.

Multiple contexts can be created, and the appropriate context is referenced on the route server by the neighbors assigned to use that context (in the **neighbor route-server-client** command). Thus, multiple neighbors sharing the same policy can share the same route server context.

Three Stages of Filtering on a Route Server Client

With the introduction of route server context, there are now three stages of route filtering that can be applied to a route server client, as shown in the figure below. The three stages of filtering are described below the figure. You can apply all, none, or any combination of the three filtering methods to a route server client. In the figure, the decreasing arrow sizes symbolize that potentially fewer routes might pass each filter than entered the filter.

Figure 68: Route Server Filtering in Three Stages



1. As shown in the figure above beginning at the left, when incoming eBGP updates arrive from a route server client, the system will apply inbound route filters for a route server client the same way it does for a non-route-server client (configured with the **neighbor route-map in** command). All routes permitted by the client's inbound filtering are installed in the global BGP table for the appropriate address family, as usual, and anything else is dropped.

2. If any route server contexts have been configured with flexible policy using the **import-map** command, the best path from among the subset of matching routes is imported into the virtual table for the contexts. Route server clients associated with a context will then override any routes from the global BGP table with customized routes from the context's virtual table when generating updates.

3. A route server client's outbound filtering policies (configured with the **neighbor route-map out** command) will be applied to the global updates that do not have customized policy, and the outbound filtering policies are also applied to any updates generated from the route server context's virtual table.

How to Configure a BGP Route Server

Configure a Route Server with Basic Functionality

Perform this task to configure a BGP router as route server for IPv4 or IPv6. This task will enable the basic route server functionality for nexthop, AS-path, and MED transparency.

Note This task does not enable flexible policy handling. To enable flexible policy handling, see the Configure a Route Server with Flexible Policy Handling, on page 728.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **neighbor** {*ipv4-address*| *ipv6-address*} **remote-as** *remote-as-number*
5. **address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**}
6. **neighbor** {*ipv4-address*| *ipv6-address*} **activate**
7. **neighbor** {*ipv4-address*| *ipv6-address*} **route-server-client**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 900`	Configures a BGP routing process.
Step 4	**neighbor** {*ipv4-address*| *ipv6-address*} **remote-as** *remote-as-number* **Example:**	Adds an entry to the BGP neighbor table.

	Command or Action	Purpose
	`Router(config-router)# neighbor 10.0.0.1 remote-as 100`	
Step 5	**address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**} **Example:** `Router(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure a routing session using IPv4 or IPv6 unicast or multicast address prefixes.
Step 6	**neighbor** {*ipv4-address*| *ipv6-address*} **activate** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 7	**neighbor** {*ipv4-address*| *ipv6-address*} **route-server-client** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 route-server-client`	Configures the BGP neighbor at the specified address to be a route server client.
Step 8	**end** **Example:** `Router(config-router-af)# end`	Ends the current configuration and returns to privileged EXEC mode.

Configure a Route Server Client To Receive Updates

In the prior task, you configured a route server. A route server does not put its own AS number in the AS-path; there is AS-path transparency. This means the route server client will receive updates in which the first AS number in the AS-path is not the sending router's AS number.

By default, a router denies an update received from an eBGP peer that does not list its AS number at the beginning of the AS-path in an incoming update. Therefore, you must disable that behavior on the client in order for the client to receive the updates. To do so, perform this task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **no bgp enforce-first-as**
5. **neighbor** {*ipv4-address*\| *ipv6-address*} **remote-as** *remote-as-number*
6. **address-family** {**ipv4** \| **ipv6**} { **unicast** \| **multicast**}
7. **neighbor** {*ipv4-address*\| *ipv6-address*} **activate**
8. **exit-address-family**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 900`	Configures a BGP routing process.
Step 4	**no bgp enforce-first-as** **Example:** `Router(config-router)# no bgp enforce-first-as`	Disables requirement that an update received from an eBGP peer list its AS number at the beginning of the AS_PATH. • By default, a router is configured to deny an update received from an external BGP (eBGP) peer that does not list its autonomous system number at the beginning of the AS_PATH in the incoming update. • In order to receive updates from the route server, which will not have its AS first in the AS_PATH, specify **no bgp enforce-first-as**to disable the enforcement.
Step 5	**neighbor** {*ipv4-address*| *ipv6-address*} **remote-as** *remote-as-number* **Example:** `Router(config-router)# neighbor 10.0.0.1 remote-as 100`	Adds an entry to the BGP neighbor table.
Step 6	**address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**} **Example:** `Router(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure a routing session using IPv4 or IPv6 unicast or multicast address prefixes.
Step 7	**neighbor** {*ipv4-address*| *ipv6-address*} **activate** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 8	**exit-address-family** **Example:**	Exits address family configuration mode.

	Command or Action	Purpose
	`Router(config-router-af)# exit-address-family`	

Configure a Route Server with Flexible Policy Handling

Perform this task if you need your BGP route server to support a customized, flexible policy in addition to basic route server functionality.

In order to configure flexible policy handling, create a route server context, which includes an import map. The import map references a standard route map.

In this particular configuration task, the policy is based on autonomous system number, so the **match as-path** command is used. The actual AS number is identified in the **ip as-path access-list** command. You may match on nexthop, AS path, communities, and extended communities.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **route-server-context** *context-name*
5. **description** *string*
6. **address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**}
7. **import-map** *route-map-name*
8. **exit-address-family**
9. **exit-route-server-context**
10. **exit**
11. **ip as-path access-list** *access-list-number* {**permit**| **deny**} *regexp*
12. **route-map** *route-map-name* [**permit** | **deny**] *sequence-number*
13. **match as-path** *access-list-number*
14. **exit**
15. **router bgp** *autonomous-system-number*
16. **neighbor** {*ipv4-address*| *ipv6-address*} **remote-as** *remote-as-number*
17. **address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**}
18. **neighbor** {*ipv4-address*| *ipv6-address*} **activate**
19. **neighbor** {*ipv4-address*| *ipv6-address*} **route-server-client context** *ctx-name*
20. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 900`	Configures a BGP routing process.
Step 4	**route-server-context** *context-name* **Example:** `Router(config-router)# route-server-context ONLY_AS27_CONTEXT`	Creates a route server context. • In this example, a context named ONLY_AS27_CONTEXT is created.
Step 5	**description** *string* **Example:** `Router(config-router-rsctx)# description Permit only routes with AS 27 in AS path.`	(Optional) Allows you to describe the context. • Up to 80 characters are allowed.
Step 6	**address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**} **Example:** `Router(config-router-rsctx)# address-family ipv4 unicast`	Enters address family configuration mode to configure a routing session using IPv4 or IPv6 unicast or multicast address prefixes.
Step 7	**import-map** *route-map-name* **Example:** `Router(config-router-rsctx-af)# import-map only_AS27_routemap`	Configures flexible policy handling by using the route map that you will create in Step 12 to control which routes will be added to the route server client virtual table.
Step 8	**exit-address-family** **Example:** `Router(config-router-rsctx-af)# exit-address-family`	Exits address family configuration mode.
Step 9	**exit-route-server-context** **Example:** `Router(config-router-rsctx)# exit-route-server-context`	Exits route server context configuration mode.
Step 10	**exit** **Example:**	Exits router configuration mode.

	Command or Action	Purpose
	`Router(config-router)# exit`	
Step 11	**ip as-path access-list** *access-list-number* {**permit**| **deny**} *regexp* **Example:** `Router(config)# ip as-path access-list 5 permit 27`	Configures an AS path filter using a regular expression. • The **ip as-path** command is not necessarily the command you have to use. Determine what policy you want to create.
Step 12	**route-map** *route-map-name* [**permit** | **deny**] *sequence-number* **Example:** `Router(config)# route-map only_AS27_routemap permit 10`	Defines whether AS paths that match the subsequent **match as-path**command will be permitted or denied in the route map. • Use the same *route-map-name* that you specified in the **import-map** command above.
Step 13	**match as-path** *access-list-number* **Example:** `Router(config-route-map)# match as-path 5`	Identifies an access list that determines which AS paths are matched and become part of the route map configured in the prior step. • This particular example references the *access-list-number* configured in the **ip as-path access-list** command. • The **match as-path** command is not necessarily the command you have to use. Determine what policy you want to use. • You may match on nexthop, AS path, communities, and extended communities.
Step 14	**exit** **Example:** `Router(config-route-map)# exit`	Exits route map configuration mode.
Step 15	**router bgp** *autonomous-system-number* **Example:** `Router(config)# router bgp 900`	Configures a BGP routing process.
Step 16	**neighbor** {*ipv4-address*| *ipv6-address*} **remote-as** *remote-as-number* **Example:** `Router(config-router)# neighbor 10.0.0.1 remote-as 500`	Adds an entry to the BGP neighbor table.

	Command or Action	Purpose
Step 17	**address-family** {**ipv4** | **ipv6**} { **unicast** | **multicast**} **Example:** `Router(config-router)# address-family ipv4 unicast`	Enters address family configuration mode to configure a routing session using IPv4 or IPv6 unicast or multicast address prefixes.
Step 18	**neighbor** {*ipv4-address*| *ipv6-address*} **activate** **Example:** `Router(config-router-af)# neighbor 10.0.0.1 activate`	Enables the exchange of information with a BGP neighbor.
Step 19	**neighbor** {*ipv4-address*| *ipv6-address*} **route-server-client context** *ctx-name* **Example:** `Router(config-router-af)# neighbor 10.0.0.1 route-server-client context ONLY_AS27_CONTEXT`	Configures the BGP neighbor at the specified address to be a route server client. • In this example, the route server client at this specified address is assigned to the context called ONLY_AS27_CONTEXT.
Step 20	**end** **Example:** `Router(config-router-af)# end`	Ends the current configuration and returns to privileged EXEC mode.

Displaying BGP Route Server Information and Troubleshooting Route Server

From privileged EXEC mode, perform either of the steps in this task on a BGP route server to see information about the route server.

On a BGP route server client (not the route server), you can use the **show ip bgp ipv4 unicast** or **show ip bgp ipv6 unicast**command to display routing information.

SUMMARY STEPS

1. **enable**
2. **show ip bgp** {**ipv4** \| **ipv6**} **unicast route-server** {**all** \| {**context** *context-name*}} [**summary**]
3. **debug ip bgp route-server** {**client** \| **context** \| **event** \| **import** \| **policy**} [**detail**]

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**show ip bgp** {**ipv4** | **ipv6**} **unicast route-server** {**all** | {**context** *context-name*}} [**summary**] **Example:** `Router#`	Displays the paths chosen for a particular route server context, which migh include the global bestpath, an overriding policy path, or a suppressed path.
Step 3	**debug ip bgp route-server** {**client** | **context** | **event** | **import** | **policy**} [**detail**] **Example:** `Router# debug ip bgp route-server client`	Turns on debugging for BGP route server. **Caution** The **detail** keyword is used for more complex issues and should only be turned on when debugging with a Cisco representative.

Configuration Examples for BGP Route Server

Example BGP Route Server with Basic Functionality

In the following example, the neighbor at 10.0.0.1 is a route server client.

```
router bgp 65000
 neighbor 10.0.0.1 remote-as 100
 neighbor 10.0.0.5 remote-as 500
 address-family ipv4 unicast
 neighbor 10.0.0.1 activate
 neighbor 10.0.0.1 route-server-client
!
```

Example BGP Route Server Context for Flexible Policy (IPv4 Addressing)

In the following example, the local router is a BGP route server. Its neighbors at 10.10.10.12 and 10.10.10.13 are its route server clients. A route server context named ONLY_AS27_CONTEXT is created and applied to the neighbor at 10.10.10.13. The context uses an import map that references a route map named only_AS27_routemap. The route map matches routes permitted by access list 27. Access list 27 permits routes that have 27 in the AS path.

```
router bgp 65000
   route-server-context ONLY_AS27_CONTEXT
      address-family ipv4 unicast
         import-map only_AS27_routemap
      exit-address-family
   exit-route-server-context
   !
   neighbor 10.10.10.12 remote-as 12
   neighbor 10.10.10.12 description Peer12
   neighbor 10.10.10.13 remote-as 13
   neighbor 10.10.10.13 description Peer13
   neighbor 10.10.10.21 remote-as 21
   neighbor 10.10.10.27 remote-as 27
   !
   address-family ipv4
```

```
    neighbor 10.10.10.12 activate
    neighbor 10.10.10.12 route-server-client
    neighbor 10.10.10.13 activate
    neighbor 10.10.10.13 route-server-client context ONLY_AS27_CONTEXT
    neighbor 10.10.10.21 activate
    neighbor 10.10.10.27 activate
  exit-address-family
!
ip as-path access-list 27 permit 27
!
route-map only_AS27_routemap permit 10
  match as-path 27
!
```

Example Using Show Commands to See That Route Server Context Routes Overwrite Normal Bestpath

In the following output, a BGP route server has two routes from AS 21 that have been selected as best:

```
Route-Server# show ip bgp ipv4 unicast
BGP table version is 31, local router ID is 100.100.100.100
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
            r RIB-failure, S Stale, m multipath, b backup-path, x best-external, f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 1.1.1.1/32       10.10.10.21             23             0 21 ?
*                   10.10.10.27            878             0 27 89 ?
*  100.1.1.1/32     10.10.10.27            878             0 27 89 ?
*>                  10.10.10.21             23             0 21 ?
```

For Peer12, which has been configured as a route-server client, but not associated with any context, the bestpath is advertised in the following output. Note that AS-path, MED, and nexthop transparency have been maintained; the routes look as if they had not passed through the route server.

```
Peer12# show ip bgp ipv4 unicast

BGP table version is 31, local router ID is 10.10.10.12
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
            r RIB-failure, S Stale, m multipath, b backup-path, x best-external, f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
*> 1.1.1.1/32       10.10.10.21             23             0 21 ?
*> 100.1.1.1/32     10.10.10.21             23             0 21 ?
```

Peer13 has also been configured as a route-server client, and it has been associated with a context named ONLY_AS27_CONTEXT. The context references a route map that permits only routes that contain AS 27 in the AS path. This means that the route-server should not send any routes to Peer13 unless they contains AS 27. In our scenario, the route server indeed sends the routes learned via AS 27, even though the routes learned via AS 21 are marked as best. The output below demonstrates that the normal best path was overriden by the best path based on policy. Again, MED, as-path, and nexthop transparency have been maintained.

```
Peer13# show ip bgp ipv4 unicast

BGP table version is 25, local router ID is 10.10.10.13
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
            r RIB-failure, S Stale, m multipath, b backup-path, x best-external, f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
```

```
*> 1.1.1.1/32       10.10.10.27            878              0 27 89 ?
*> 100.1.1.1/32     10.10.10.27            878              0 27 89 ?
```

Example BGP Route Server Context with No Routes Satisfying the Policy

It is possible that, due to policy, no routes are sent to a client even though paths exist. For instance, if we take the prior example and change ONLY_AS27_CONTEXT to ONLY_AS100_CONTEXT, no paths would satisfy this policy and no routes will be sent to the client. The following is the configuration and resulting show output:

```
Route-Server# show run | begin router bgp
router bgp 1
   route-server-context ONLY_AS100_CONTEXT
     !
     address-family ipv4 unicast
        import-map only_AS100_routemap
     exit-address-family
   exit-route-server-context
   !
   neighbor 10.10.10.13 remote-as 13
   neighbor 10.10.10.13 description Peer13
   neighbor 10.10.10.21 remote-as 21
   neighbor 10.10.10.27 remote-as 27
   !
      address-family ipv4
         neighbor 10.10.10.13 activate
         neighbor 10.10.10.13 route-server-client context ONLY_AS100_CONTEXT
         neighbor 10.10.10.21 activate
         neighbor 10.10.10.27 activate
   exit-address-family

!
ip as-path access-list 100 permit 100
!
!
route-map only_AS100_routemap permit 10
   match as-path 100
!
```

Because no routes satisfy the policy, no routes appear in the table of Peer13:

```
Peer13# show ip bgp ipv4 unicast
```

Example BGP Route Server Context for Flexible Policy (IPv6 Addressing)

In the following example under address-family IPv6, the local router is a BGP route server. Its neighbors at 2001:DB8:1::112 and 2001:DB8:1::113 are its route server clients. A route server context named ONLY_AS27_CONTEXT is created and applied to the neighbor at 2001:DB8:1::113. The context uses an import map that references a route map named only_AS27_routemap. The route map matches routes permitted by access list 27. Access list 27 permits routes that have 27 in the AS path.

```
Route-Server# show run | begin router bgp
router bgp 1
   route-server-context ONLY_AS27_CONTEXT
      address-family ipv6 unicast
         import-map only_AS27_routemap
      exit-address-family
   exit-route-server-context
```

```
  !
  neighbor 2001:DB8:1::112 remote-as 12
  neighbor 2001:DB8:1::112 description Peer12
  neighbor 2001:DB8:1::113 remote-as 13
  neighbor 2001:DB8:1::113 description Peer13
 !
  address-family ipv6
     neighbor 2001:DB8:1::112 activate
     neighbor 2001:DB8:1::112 route-server-client
     neighbor 2001:DB8:1::113 activate
     neighbor 2001:DB8:1::113 route-server-client context ONLY_AS27_CONTEXT
 exit-address-family
!
ip as-path access-list 27 permit 27
!
route-map only_AS27_routemap permit 10
   match as-path 27
!
Route-Server#show ip bgp ipv6 unicast route-server all summary

Route server clients without assigned contexts:
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2001:DB8:1::112 4           12      19      19        4    0    0 00:12:50        2
Route server clients assigned to context ONLY_AS27_CONTEXT:
Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2001:DB8:1::113 4           13      23      22        4    0    0 00:16:23        2
```

For Peer12, which has been configured as a route-server client, but not associated with any context, the bestpath is advertised. Note that AS-path, MED, and nexthop transparency have been maintained; the routes look as if they had not passed through the route server.

```
Peer12# show ip bgp ipv6 unicast
BGP table version is 9, local router ID is 2.2.2.2
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
            r RIB-failure, S Stale, m multipath, b backup-path, x best-external, f RT-Filter
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric LocPrf Weight Path
* 2001:DB8:1::/64  2001:DB8::113            0             0 13 ?
*>                  ::                       0         32768 ?
* 2001:DB8:2::/64  2001:DB8::113            0             0 13 ?
*>                  ::                       0         32768 ?
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	*Cisco IOS IP Routing: BGP Command Reference*
BGP configuration tasks	*IP Routing: BGP Configuration Guide, Cisco IOS XE Release 3S*

MIBs

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

Technical Assistance

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

Feature Information for BGP Route Server

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 55: Feature Information for BGP Route Server

Feature Name	Releases	Feature Information
BGP Route Server	Cisco IOS XE Release 3.3S 15.2(3)T	BGP route server is a feature designed for internet exchange (IX) operators that provides an alternative to full eBGP mesh peering among the service providers who have a presence at the IX. The route server provides eBGP route reflection with customized policy support for each service provider. That is, a route server context can override the normal BGP best path for a prefix with a different path based on a policy, or suppress all paths for a prefix and not advertise the prefix. The BGP route server provides reduced configuration complexity and reduced CPU and memory requirements on each border router. The route server also reduces overhead expense incurred by individualized peering agreements. The following commands were introduced: • **debug ip bgp route-server** • **description** (route server context) • **exit-route-server-context** • **import-map** • **neighbor route-server-client** • **route-server-context** • **show ip bgp unicast route-server**

CHAPTER 43

BGP Diverse Path Using a Diverse-Path Route Reflector

The BGP Diverse Path Using a Diverse-Path Route Reflector feature allows Border Gateway Protocol (BGP) to distribute an alternative path other than the best path between BGP speakers when route reflectors are deployed. This feature is meant to provide path diversity within an autonomous system (AS), within a single cluster only. That is, a route reflector is allowed to advertise the diverse path to its client peers only.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see the 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 http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Prerequisites for BGP Diverse Path Using a Diverse-Path Route Reflector

You should understand the BGP Best External feature.

Restrictions for BGP Diverse Path Using a Diverse-Path Route Reflector

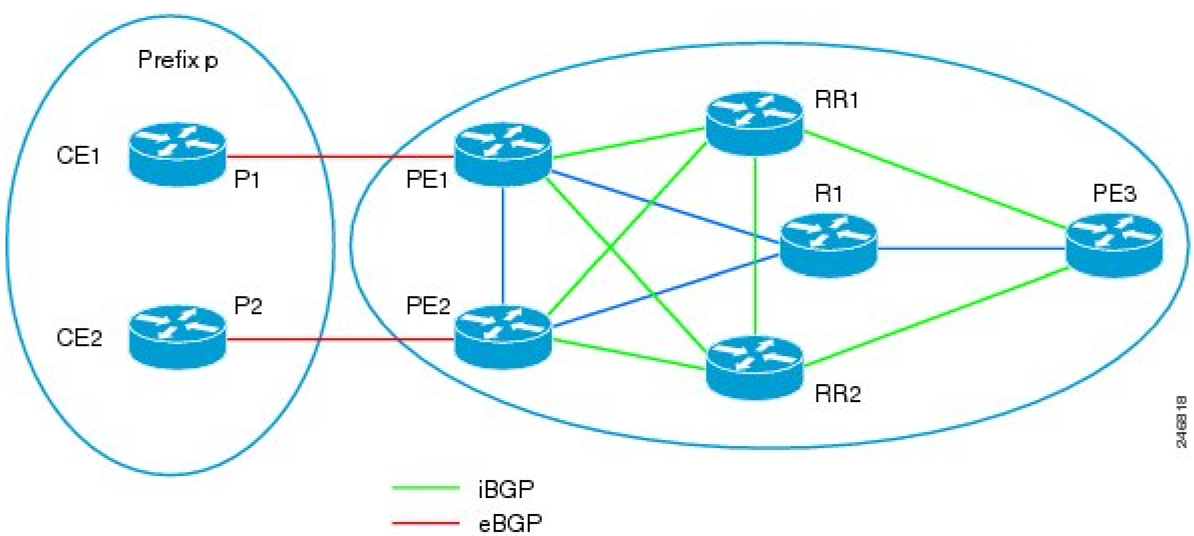
- A diverse path can be configured on a route reflector only.
- Only one shadow route reflector is allowed per existing route reflector, which will calculate one additional best path (the second best path). That is, only one additional plane (topology) is configured.
- Path diversity is configured within an AS, within a single route reflector cluster. That is, the route reflector will advertise the diverse path to its route reflector client peers only.
- Diverse path functionality is not supported on a route server.

Information About BGP Diverse Path Using a Diverse-Path Reflector

Limitation that a BGP Diverse Path Overcomes

As a path vector routing protocol, BGP-4 requires a router to advertise to its neighbors only the best path for a destination. However, multiple paths to the same destination would allow mechanisms that can improve resilience, quickly recover from failures, and load balance, for example.

The use of route reflectors is one of the main reasons for poor path diversity within an autonomous system (AS). In a network with route reflectors, even if a prefix is learned from multiple egress routers, the route reflector reflects only the best path to its clients. The figure below shows how deploying route reflectors might reduce path diversity in an AS, even when the BGP Best External feature is deployed.



In the figure above, P1 and P2 are diverse paths for prefix p. Assume Path 2 (P2) has a lower MED and higher local preference than P1. The BGP Best External feature on PE1 will make sure that P1 is propagated to the route reflectors, regardless of P2 having a lower MED and higher local preference. The route reflectors will have path diversity; they will learn both P1 and P2 with different exit points PE1 and PE2 (assuming that PE1 and PE2 have the **set ip next-hop self** command configured). However, both route reflectors select the best path as P2 due to its lower MED/higher local preference and advertise it to PE3. PE3 will not learn P1 (that is, PE3 will not learn about existing path diversity).

The BGP Diverse Path Using a Diverse-Path Route Reflector feature is a way to resolve that limitation and achieve path diversity.

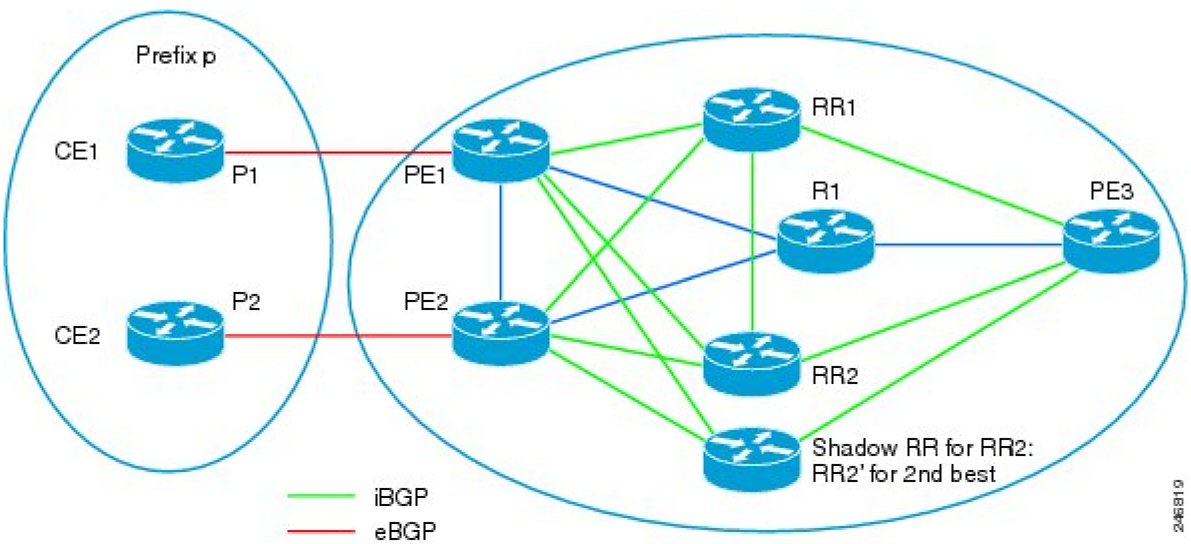
BGP Diverse Path Using a Diverse-Path Route Reflector

The BGP Diverse Path Using a Diverse-Path Route Reflector feature overcomes the lack of path diversity in an AS containing route reflectors. This feature is meant to provide path diversity within an AS, within a single cluster only. That is, a route reflector is allowed to advertise the diverse path to its client peers only.

For each route reflector in the AS, a *shadow route reflector* is added to distribute the *second best path*, also known as the *diverse path*. The figure below shows the shadow route reflector for RR2. The shadow route reflector improves path diversity because PE3 can now learn both P1 (from RR1/RR2) and learn P2 from the shadow route reflector.

Note The primary route reflector and shadow route reflector must have the exact same connections (physical/control plane) to the rest of the routers in the network.

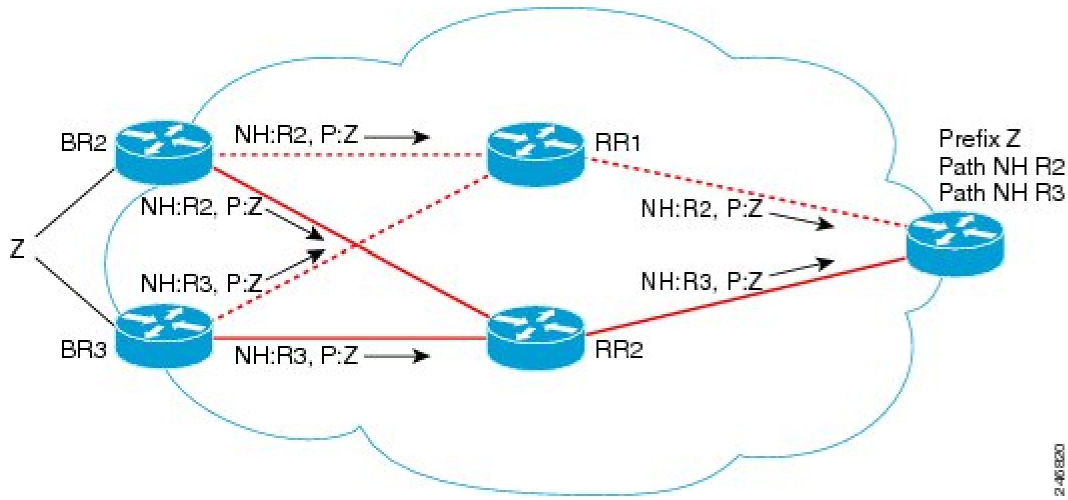
Shadow route reflectors can be both control plane route reflectors and data plane route reflectors.



The figure below shows a diverse path in greater detail, indicating the next hops:

- BR2 announces to RR1 and shadow RR2 that R2 (BR2) is the Next Hop for those who want to reach Prefix Z. Likewise, BR3 announces to RR1 and shadow RR2 that R3 (BR3) is the Next Hop for those who want to reach Prefix Z
- RR1 sends a packet to BR1 announcing that the Next Hop is R2 if BR1 wants to reach Prefix Z. The second best path (or diverse path) comes from shadow RR2, which sends a packet to BR1 announcing that the Next Hop is R3 if BR1 want to reach Prefix Z.

- At BR1 (far right), we see there are two (diverse) paths to Prefix Z.



Triggers to Compute a BGP Diverse Path

Computation of a diverse path per address family is triggered by any of the following commands:

- **bgp additional-paths install**
- **bgp additional-paths select**
- **maximum-paths ebgp**
- **maximum-paths ibgp**

The **bgp additional-paths install** command will install the type of path that is specified in the **bgp additional-paths select** command. If the **bgp additional-paths select** command specifies both keyword options (**best-external** and **backup**), the system will install a backup path.

The **maximum-paths ebgp** and **maximum-paths ibgp**commands trigger a multipath computation, and multipaths are automatically installed as primary paths.

On the other hand, the **bgp additional-paths install** command triggers computation of a backup path or best-external path.

If the **bgp additional-paths select** command is not configured, the **bgp additional-paths install**command will trigger both computation and installation of a backup path (as is done with the BGP PIC feature).

IGP Metric Check

Disabling the Interior Gateway Protocol (IGP) metric check and configuring the BGP Diverse Path feature are independent of each other. One does not imply the other. That is, configuring **bgp bestpath igp-metric ignore** does not imply that the BGP Diverse Path feature is enabled. Conversely, enabling the BGP Diverse Path feature might not require that **bgp bestpath igp-metric ignore** be configured (because, for example, the route reflector and shadow route reflector are co-located).

The **bgp bestpath igp-metric ignore** command can be configured at route reflectors and provider edges (PEs).

Note Per-VRF functionality for the **bgp bestpath igp-metric ignore** command is not supported. If you use it anyway, it is at your own risk.

Route Reflector Determination

If a router's configuration includes either one of the following commands, the router is a route reflector:

- **bgp cluster-id**
- **neighbor route-reflector-client**

How to Configure a BGP Diverse-Path Route Reflector

Determining Whether You Need to Disable the IGP Metric Check

Before you configure a shadow route reflector in order to get a BGP diverse path, determine whether you need to disable the IGP metric check. The IGP metric is a configurable value indicating physical distance, and is used by an Interior Gateway Protocol, such as Open Shortest Path First (OSPF), Enhanced Interior Gateway Routing Protocol (EIGRP), or Routing Information Protocol (RIP). A smaller IGP metric is preferred over a larger IGP metric.

The locations of the route reflector and shadow route reflector determine whether or not you need to disable the IGP metric check, as follows:

- When the route reflector and shadow route reflector are colocated—They have the same IP subnetwork address and are connected to the Ethernet switch with different links. Failure of such a link is equivalent to the route reflector going down. When RRs are colocated, their IGP metrics cannot be different from each other; and therefore there is no need to disable the IGP metric check during the best path calculation at any route reflector. Because there is no need to disable the IGP metric check, the first plane route reflectors do not need to be upgraded to Cisco IOS XE Release 3.4S.
- When the shadow route reflector is in a different IGP place from the route reflector (it is not colocated with its best path route reflector)--In this case, the IGP metric check is ignored on both the best path route reflector and shadow route reflector when the best path and second best path are being calculated. The IGP metric check must be disabled on the primary route reflector by configuring the **bgp bestpath igp-metric ignore** command. This command is available beginning with Cisco IOS XE Release 3.4S, which means you need to upgrade to that release.

Configuring the Route Reflector for BGP Diverse Path

Perform this task to configure a route reflector for the BGP Diverse Path feature. This task specifies the IPv4 address family, but other address families are also supported.

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**
6. **neighbor** *ip-address* **activate**
7. **maximum-paths ibgp** *number-of-paths*
8. **bgp bestpath igp-metric ignore**
9. **bgp additional-paths select** [**backup**]
10. **bgp additional-paths install**
11. **neighbor** *ip-address* **route-reflector-client**
12. **neighbor** *ip-address* **advertise diverse-path** [**backup**] [**mpath**]
13. **end**
14. **show ip bgp neighbor** *ip-address* **advertised-routes**

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 1`	Enters router configuration mode for the BGP routing process.
Step 4	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 10.1.1.1 remote-as 1`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 5	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the address family and enters address family configuration mode. • Supported address families are IPv4 unicast, VPNv4 unicast, IPv6 unicast, VPNv6 unicast, IPv4+label, and IPv6+label.
Step 6	**neighbor** *ip-address* **activate** **Example:**	Enables the exchange of information with a BGP neighbor.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 10.1.1.1 activate`	
Step 7	**maximum-paths ibgp** *number-of-paths* **Example:** `Device(config-router-af)# maximum-paths ibgp 4`	Controls the maximum number of parallel Internal BGP (IBGP) routes that can be installed in a routing table.
Step 8	**bgp bestpath igp-metric ignore** **Example:** `Device(config-router-af)# bgp bestpath igp-metric ignore`	Configures the system to ignore the Interior Gateway Protocol (IGP) metric during BGP best path selection.
Step 9	**bgp additional-paths select** [**backup**] **Example:** `Device(config-router-af)# bgp additional-paths select backup`	Configures the system to calculate a second BGP best path.
Step 10	**bgp additional-paths install** **Example:** `Device(config-router-af)# bgp additional-paths install`	Enables BGP to calculate a backup path for a given address family and to install it into the routing information base (RIB) and Cisco Express Forwarding (CEF).
Step 11	**neighbor** *ip-address* **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 10.1.1.1 route-reflector-client`	Configures the router as a BGP route reflector and configures the specified neighbor as its client.
Step 12	**neighbor** *ip-address* **advertise diverse-path** [**backup**] [**mpath**] **Example:** `Device(config-router-af)# neighbor 10.1.1.1 advertise diverse-path backup`	(Optional) Configures a neighbor to receive the diverse path in an advertisement.
Step 13	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits address family configuration mode and returns to privileged EXEC mode.
Step 14	**show ip bgp neighbor** *ip-address* **advertised-routes** **Example:** `Device# show ip bgp neighbor 10.1.1.1 advertised-routes`	(Optional) Displays the routes advertised to the specified neighbor.

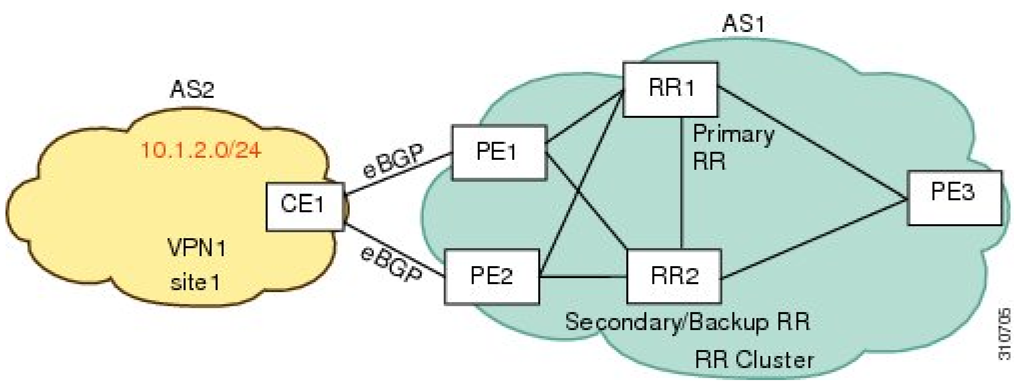
Configuration Examples for BGP Diverse Path Using a Diverse-Path Route Reflector

Example: Configuring BGP Diverse Path Where Additional Path Is the Backup Path

Diverse path functionality is contained within a single cluster; that is, only the clients of a route reflector can be configured to advertise the diverse path. A diverse path is advertised to the clients of a route reflector only if the client is configured to get the additional path.

A shadow route reflector can be added to calculate and advertise the additional path, or an existing route reflector can be configured to calculate and advertise the additional path. In the figure below, instead of adding a shadow route reflector, RR2 (the existing backup RR) is configured to calculate the additional path and advertise it to a particular neighbor.

In the figure below, assume that from the route reflectors, the path to CE1 via PE1 is preferred over the path via PE2. Without the diverse path feature, both route reflectors will advertise to PE3 that the path to CE1 is via PE1. If the connection between RR1 and PE1 fails (or the path between PE1 and CE1 fails), there is no other path.



In the following configuration example based on the figure above, RR2 is configured with an additional path, which is a backup path.

If RR1 and RR2 are not colocated, you must configure the **bgp bestpath igp-metric ignore** command before the additional path is calculated. (If RR1 and RR2 are colocated, do not configure that command.)

The **bgp additional-paths select backup** command triggers calculation of the backup path at RR2, which is the path via PE2.

The **bgp additional-paths install** command installs the backup path if RR2 is in the forwarding plane. (Do not configure this command if RR2 is in the control plane.)

The address of PE3 is 10.1.1.1, and that address is used in the **neighbor advertise diverse-path backup** command on RR2. This command triggers advertisement of the backup path to PE3. PE3 will learn the best path, (which is the path via PE1) from RR1, and it will learn the backup path from RR2.

RR2

```
router bgp 1
 neighbor 10.1.1.1 remote-as 1
 address-family ipv4 unicast
 neighbor 10.1.1.1 activate
 maximum-paths ibgp 4
 bgp bestpath igp-metric ignore
 bgp additional-paths select backup
 bgp additional-paths install
 neighbor 10.1.1.1 route-reflector-client
 neighbor 10.1.1.1 advertise diverse-path backup
```

Example: Configuring BGP Diverse Path Where Additional Path Is the Multipath

In the following example based on the figure above, assume that paths toward CE1 via PE1 and PE2 are multipaths. The **maximum-paths ibgp** command will trigger calculation of multipaths.

The address of PE3 is 10.1.1.1, and that address is used in the **neighbor advertise diverse-path mpath** command on RR2. This command will trigger advertisement of the multipath, that is, the second best path, to PE3. PE3 will learn the best path, path via PE1 from RR1, and will learn second best path from RR2.

RR2

```
router bgp 1
 neighbor 10.1.1.1 remote-as 1
 address-family ipv4 unicast
 neighbor 10.1.1.1 activate
 maximum-paths ibgp 4
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 route-reflector-client
 neighbor 10.1.1.1 advertise diverse-path mpath
```

Example: Configuring BGP Diverse Path Where Both Multipath and Backup Path Calculations Are Triggered

The following example is based on the figure above. The **maximum-paths ibgp** command will trigger calculation of multipaths. When both multipath and backup path calculations are triggered, the backup path and the second multipath (which is the second best path) are the same paths and it will be installed as the active path, regardless of whether the route reflector is in the control plane or forwarding plane.

The address of PE3 is 10.1.1.1, and that address is used in the **neighbor advertise diverse-path backup mpath** command on RR2. This command causes RR2 to advertise the second best path, which is the second multipath, to PE3.

RR2

```
router bgp 1
 neighbor 10.1.1.1 remote-as 1
 address-family ipv4 unicast
 neighbor 10.1.1.1 activate
 maximum-paths ibgp 4
 bgp additional-paths select backup
 neighbor 10.1.1.1 remote-as 1
```

```
 neighbor 10.1.1.1 route-reflector-client
 neighbor 10.1.1.1 advertise diverse-path backup mpath
```

Example: Configuring Triggering Computation and Installation of a Backup Path

When the **bgp additional-paths install** command is configured without configuring **bgp additional-paths select backup**, the former command will trigger both computation and installation of the backup path (as it is with the existing BGP PIC feature).

The address of PE3 is 10.1.1.1, and that address is used in the **neighbor advertise diverse-path backup** command on RR2. This command will trigger advertisement of a backup path to PE3. PE3 will learn the best path, a path via PE1 from RR1, and it will learn a backup path from RR2.

RR2

```
router bgp 1
 neighbor 10.1.1.1 remote-as 1
 address-family ipv4 unicast
 neighbor 10.1.1.1 activate
 maximum-paths ibgp 4
 bgp additional-paths install
 neighbor 10.1.1.1 remote-as 1
 neighbor 10.1.1.1 route-reflector-client
 neighbor 10.1.1.1 advertise diverse-path backup
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
Configuring BGP Best External Path on a Route Reflector for Intercluster	BGP Best External module
BGP configuration tasks	Cisco IOS XE IP Routing: BGP Configuration Guide

Standards

Standard	Title
draft-ietf-grow-diverse-bgp-path-dist-02.txt	*Distribution of Diverse BGP Paths*

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To 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 4271	A Border Gateway Protocol 4 (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 Diverse Path Using a Diverse-Path Route Reflector

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

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

Table 56: Feature Information for BGP Diverse Path Using a Diverse-Path Route Reflector

Feature Name	Releases	Feature Information
BGP Diverse Path Using a Diverse-Path Route Reflector	15.2(3)T	This feature allows BGP to distribute an alternative path other than the best path between BGP speakers when route reflectors are deployed. The following commands were introduced: • **bgp additional-paths select** • **bgp bestpath igp-metric ignore** • **debug ip bgp igp-metric ignore** • **neighbor advertise best-external** • **neighbor advertise diverse-path**

CHAPTER 44

BGP-VRF-Aware Conditional Advertisement

The Border Gateway Protocol (BGP) VRF-Aware Conditional Advertisement feature provides additional control of the advertisement of routes and extends this control to within a virtual routing and forwarding (VRF) instance.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information 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 http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Information About BGP VRF-Aware Conditional Advertisement

VRF-Aware Conditional Advertisement

The Border Gateway Protocol (BGP) VRF-Aware Conditional Advertisement feature provides additional control of the advertisement of routes and extends this control within a virtual routing and forwarding (VRF) instance.

BGP Conditional Advertisement

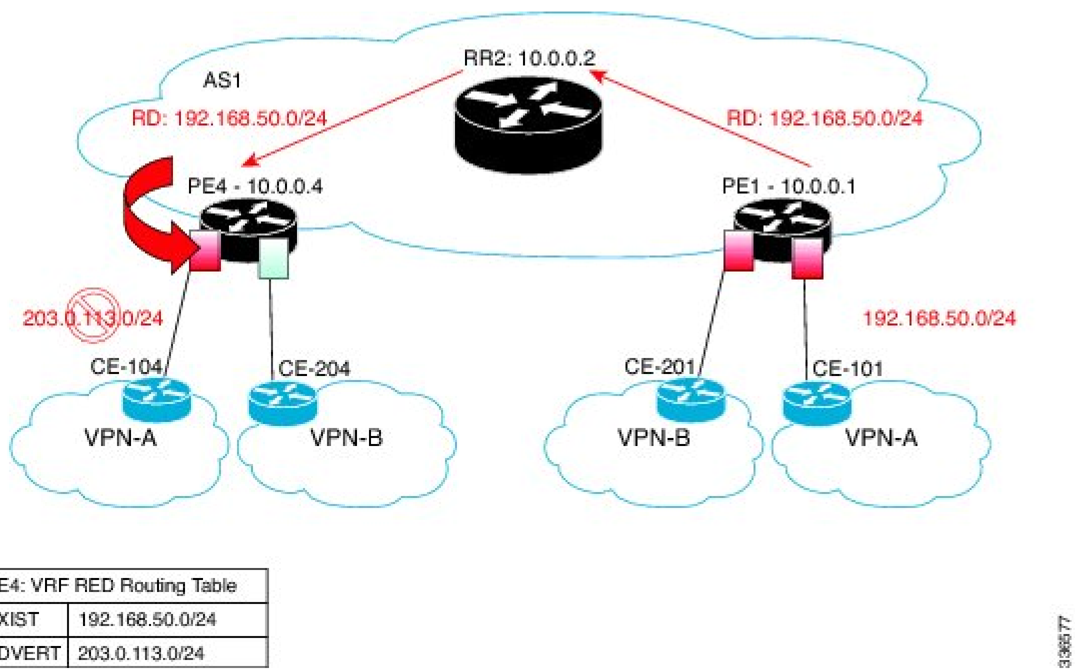
Normally, routes are propagated regardless of the existence of a different route. The BGP conditional advertisement feature uses the **exist-map**, **non-exist-map**, and the **advertise-map** keywords of the **neighbor** command in order to track routes by the route prefix. If a route prefix is not present in output of the **non-exist-map** command, then the route specified by the **advertise-map** is announced. This feature is useful for multihomed networks, in which some prefixes are advertised to one of the providers only if information from the other provider is not present (this indicates a failure in the peering session or partial reachability). The conditional BGP announcements are sent in addition to the normal announcements that a BGP router sends to its peers.

VRF-Aware Conditional Advertisement

This feature extends support for BGP VRF-aware conditional advertisement to the following address families:

- IPv4 unicast
- IPv4 unicast VRF
- IPv6 unicast
- IPv6 unicast VRF

Figure 69: VRF-Based Conditional Advertisement



The figure above shows the IPv4 prefix 192.168.50.0/24 being advertised by a remote CE101 into VRF RED on PE1. The prefix flows as a MP-BGP VPN prefix and is imported into the VRF RED on PE4. On the PE4 the conditions configured by the **exist-map** command relating to this prefix in the BGP VRF RED table becomes the condition to advertise the prefix 203.0.113.0/24 to the CE104, that is, peer-activated under the VRF RED on the PE4. This scenario assumes that 203.0.113.0/24 is in the VRF RED BGP table. If 203.0.113.0/24 is not in the table, this policy is ignored.

- If 192.168.50.0/24 exists in PE4's BGP table, then the 203.0.113.0/24 network is advertised to CE104.
- If 192.168.50.0/24 does not exist in PE4's BGP table, then the 203.0.113.0/24 network is not advertised to CE104.

How to Configure BGP VRF-Aware Conditional Advertisement

Configuring BGP VRF-Aware Conditional Advertisement

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. Enter one of the following:
   - **address-family ipv4** [**unicast**] [**vrf** *vrf-name*]
   - **address-family ipv6** [**unicast**] [**vrf** *vrf-name*]
5. **neighbor** {*ip-address* | *ipv6-address*} **remote-as** *autonomous-system-number*
6. **neighbor** {*ip-address* | *ipv6-address*} **activate**
7. **neighbor** {*ip-address* | *ipv6-address*} **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*}
8. **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.
Step 4	Enter one of the following: • **address-family ipv4** [**unicast**] [**vrf** *vrf-name*] • **address-family ipv6** [**unicast**] [**vrf** *vrf-name*] **Example:** `Device(config-router)# address-family ipv4 vrf VRFRED`	Specifies the IPv4 or IPv6 address family and enters address family configuration mode. • The **unicast** keyword specifies the IPv4 or IPv6 unicast address family. • The **vrf** keyword and *vrf-name* argument specify the name of the virtual routing and forwarding (VRF)

	Command or Action	Purpose
		instance to associate with subsequent IPv4 or IPv6 address family configuration mode commands.
Step 5	**neighbor** {*ip-address* | *ipv6-address*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.0.2.1 remote-as 104`	Adds the IP address of the neighbor in the specified autonomous system to the IPv4 or IPv6 multiprotocol BGP neighbor table of the local device.
Step 6	**neighbor** {*ip-address* | *ipv6-address*} **activate** **Example:** `Device(config-router-af)# neighbor 192.0.2.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 or IPv6 address family with the local device.
Step 7	**neighbor** {*ip-address* | *ipv6-address*} **advertise-map** *map-name* {**exist-map** *map-name* | **non-exist-map** *map-name*} **Example:** `Device(config-router-af)# neighbor 192.0.2.1 advertise-map ADV-1 exist-map EXIST-1`	Enables conditional advertisement towards a neighbor to allow the advertisement of prefixes mapped by the **advertise-map** command based on the criteria defined under **exist** or **non-exist** maps. • The **advertise-map** *map-name* keyword-argument pair specifies the name of the route map used to define the advertised routes. • The **exist-map** *map-name* keyword-argument pair specifies the condition that can be satisfied by a set of routes in the BGP table. If the condition is satisfied then the routes in the BGP table matching those specified in **advertise map** will be advertised. If the routes matching those specified in **exist-map** do not exist in the BGP table, those routes will not be advertised. • The **non-exist-map** *map-name* keyword-argument pair specifies the condition that is compared to a set of routes in the BGP table. If the routes in the **non-exist-map** are not present in the BGP table, then the routes matching those specified in **advertise map** will be advertised. If the routes matching those specified in **non-exist-map** are present in the BGP table, then the routes matching **advertise-map** will not be advertised.
Step 8	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

What to do next

To verify the configuration of the BGP VRF-Aware Conditional Advertisement feature, use the **show bgp ip neighbors** command.

Configuration Examples for BGP VRF-Aware Conditional Advertisement

Example: Configuring BGP VRF-Aware Conditional Advertisement

The following examples use the configuration in figure 1:

CE 101: The source of the prefixes

```
router bgp 101
 bgp log-neighbor-changes
 timers bgp 0 0
 neighbor 172.16.1.2 remote-as 65000
 !
 address-family ipv4
  network 21.21.21.0 mask 255.255.255.0
  network 22.22.22.22 mask 255.255.255.255
  network 31.0.0.0
  network 33.0.0.0
  network 44.0.0.0
  network 192.0.254 mask 255.255.255.0
  network 192.0.2.50
  neighbor 172.16.1.3 activate
 exit-address-family
```

PE 1

```
router bgp 65000
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 timers bgp 0 0
 neighbor 10.0.0.2 remote-as 65000
 neighbor 10.0.0.2 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family vpnv4
  neighbor 10.0.0.2 activate
  neighbor 10.0.0.2 send-community both
 exit-address-family
 !
 address-family ipv4 vrf blue
  neighbor 198.51.100.10 remote-as 201
  neighbor 198.51.100.10 activate
 exit-address-family
 !
 address-family ipv4 vrf red
  neighbor 172.16.1.2 remote-as 101
  neighbor 172.16.1.2 activate
 exit-address-family
```

PE 4

```
router bgp 65000
 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 timers bgp 0 0
 neighbor 10.0.0.2 remote-as 65000
 neighbor 10.0.0.2 update-source Loopback0
 !
 address-family ipv4
 exit-address-family
 !
 address-family vpnv4
  neighbor 10.0.0.2 activate
  neighbor 10.0.0.2 send-community extended
 exit-address-family
 !
 address-family ipv4 vrf blue
  neighbor 198.51.100.12 remote-as 204
  neighbor 198.51.100.12 activate
 exit-address-family
 !
 address-family ipv4 vrf red
  neighbor 198.51.100.3 remote-as 104
  neighbor 198.51.100.3 activate
  neighbor 198.51.100.3 advertise-map ADV-1 exist-map EXIST-1
  neighbor 198.51.100.3 advertise-map ADV-2 exist-map EXIST-2
  neighbor 198.51.100.3 advertise-map ADV-3 exist-map EXIST-3
  neighbor 198.51.100.3 advertise-map ADV-4 exist-map EXIST-4
 exit-address-family
!
ip prefix-list pl-adv-1 seq 5 permit 22.22.22.22/32
!
ip prefix-list pl-adv-2 seq 5 permit 44.0.0.0/8
!
ip prefix-list pl-adv-3 seq 5 permit 33.0.0.0/8
!
ip prefix-list pl-adv-4 seq 5 permit 128.16.16.0/24
!
ip prefix-list pl-exist-1 seq 5 permit 21.21.21.0/24
!
ip prefix-list pl-exist-2 seq 5 permit 41.0.0.0/8
!
ip prefix-list pl-exist-3 seq 5 permit 31.0.0.0/8
!
ip prefix-list pl-exist-4 seq 5 permit 192.168.50.0/24
!
route-map EXIST-4 permit 10
 match ip address prefix-list pl-exist-4
!
route-map ADV-4 permit 10
 match ip address prefix-list pl-adv-4
!
route-map EXIST-2 permit 10
 match ip address prefix-list pl-exist-2
!
route-map ADV-2 permit 10
 match ip address prefix-list pl-adv-2
!
route-map EXIST-3 permit 10
 match ip address prefix-list pl-exist-3
!
route-map ADV-3 permit 10
 match ip address prefix-list pl-adv-3
```

```
!
route-map EXIST-1 permit 10
 match ip address prefix-list pl-exist-1
!
route-map ADV-1 permit 10
 match ip address prefix-list pl-adv-1
```

Example: Verifying BGP VRF-Aware Conditional Advertisement

The following examples use the configuration in figure 1:

CE 101

```
CE101# show ip bgp all

For address family: IPv4 Unicast
BGP table version is 28, local router ID is 203.0.113.11
     Network          Next Hop            Metric LocPrf Weight Path
 *>  21.21.21.0/24     0.0.0.0                  0         32768 i
 *>  22.22.22.22/32    0.0.0.0                  0         32768 i
 *>  31.0.0.0          0.0.0.0                  0         32768 i
 *>  33.0.0.0          0.0.0.0                  0         32768 i
 *>  44.0.0.0          0.0.0.0                  0         32768 i
 *>  192.0.2.254/24    0.0.0.0                  0         32768 i
 *>  192.0.2.50        0.0.0.0                  0         32768 i
```

PE 1

```
PE1# show ip bgp all

For address family: IPv4 Unicast


For address family: VPNv4 Unicast

BGP table version is 46, local router ID is 10.0.0.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf red)
 *>  21.21.21.0/24    172.16.1.2             0             0 101 i
 *>  22.22.22.22/32   172.16.1.2             0             0 101 i
 *>  31.0.0.0         172.16.1.2             0             0 101 i
 *>  33.0.0.0         172.16.1.2             0             0 101 i
 *>  44.0.0.0         172.16.1.2             0             0 101 i
 *>  192.0.2.254/24   172.16.1.2             0             0 101 i
 *>  192.0.2.50       172.16.1.2             0             0 101 i
```

PE 4

Note The status is Withdraw for the exist-map EXIST-2 because the condition for advertisement has not been met.

```
PE4# show ip bgp all

For address family: VPNv4 Unicast

BGP table version is 82, local router ID is 10.0.0.4

     Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf red)
 *>i 21.21.21.0/24    10.0.0.1                 0    100      0 101 i
 *>i 22.22.22.22/32   10.0.0.1                 0    100      0 101 i
 *>i 31.0.0.0         10.0.0.1                 0    100      0 101 i
 *>i 33.0.0.0         10.0.0.1                 0    100      0 101 i
 *>i 44.0.0.0        10.0.0.1               0    100      0 101 I    <- missing 41.0.0.0/8

 *>i 192.0.2.254/24   10.0.0.1                 0    100      0 101 i
 *>i 192.0.2.50       10.0.0.1                 0    100      0 101 i


PE4# show ip bgp vpnv4 all neighbors 198.51.100.3
…
…
 For address family: VPNv4 Unicast
  Translates address family IPv4 Unicast for VRF red
  Session: 198.51.100.3
  BGP table version 48, neighbor version 48/0
  Output queue size : 0
  Index 3, Advertise bit 0
  3 update-group member
  Condition-map EXIST-1, Advertise-map ADV-1, status: Advertise
  Condition-map EXIST-2, Advertise-map ADV-2, status: Withdraw
  Condition-map EXIST-3, Advertise-map ADV-3, status: Advertise
  Condition-map EXIST-4, Advertise-map ADV-4, status: Advertise
  Slow-peer detection is disabled
  …
…
PE4#


PE4# show ip bgp vpnv4 all update-group

…
…
BGP version 4 update-group 3, external, Address Family: VPNv4 Unicast
  BGP Update version : 48/0, messages 0
  Condition-map EXIST-1, Advertise-map ADV-1, status: Advertise
  Condition-map EXIST-2, Advertise-map ADV-2, status: Withdraw
  Condition-map EXIST-3, Advertise-map ADV-3, status: Advertise
  Condition-map EXIST-4, Advertise-map ADV-4, status: Advertise
  Topology: red, highest version: 47, tail marker: 47
  Format state: Current working (OK, last not in list)
                Refresh blocked (not in list, last not in list)
  Update messages formatted 4, replicated 4, current 0, refresh 0, limit 1000
  Number of NLRIs in the update sent: max 3, min 0
  Minimum time between advertisement runs is 0 seconds
  Has 1 member:
   198.51.100.3
```

CE 104

Note Prefix 44.0.0.0 is missing as 41.0.0.0/8 does not appear in PE 4 to trigger the advertisement to CE 104. The state is Withdraw.

```
CE104# show ip bgp all

For address family: IPv4 Unicast

BGP table version is 45, local router ID is 198.51.100.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter,
              x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
 *>  21.21.21.0/24    104.0.0.1            0       65000 101     i
 *>  22.22.22.22/32   104.0.0.1            0       65000 101     i
 *>  31.0.0.0         104.0.0.1            0       65000 101     i
 *>  33.0.0.0         104.0.0.1            0       65000 101     i
 *>  192.0.2.254/24   104.0.0.1            0       65000 101     i
 *>  192.0.2.50       104.0.0.1            0       65000 101     i
```

Additional References for BGP VRF-Aware Conditional Advertisement

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 website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP VRF-Aware Conditional Advertisement

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 57: Feature Information for BGP VRF-Aware Conditional Advertisement

Feature Name	Releases	Feature Information
BGP VRF-Aware Conditional Advertisement	15.3(3)M	The Border Gateway Protocol (BGP) VRF-Aware Conditional Advertisement feature provides additional control of the advertisement of routes and extends this control to within a virtual routing and forwarding (VRF) instance.

CHAPTER 45

Configuring BGP Consistency Checker

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

Finding Feature Information

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

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

Information About BGP Consistency Checker

BGP Consistency Checker

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

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

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

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

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

How to Configure BGP Consistency Checker

Configure BGP Consistency Checker

SUMMARY STEPS

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

DETAILED STEPS

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

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

Configuration Examples for BGP Consistency Checker

Example: Configuring BGP Consistency Checker

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

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

Additional References

Related Documents

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

MIBs

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

Technical Assistance

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

Feature Information for BGP Consistency Checker

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

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

Table 58: Feature Information for BGP Consistency Checker

Feature Name	Releases	Feature Information
BGP Consistency Checker	15.2(3)T	The BGP Consistency Checker feature provides a way to identify three types of BGP route inconsistencies with peers: next-hop label inconsistency, RIB-out inconsistency, and aggregation inconsistency. Upon finding such inconsistency, the system sends a syslog error message and takes appropriate action if configured to do so. The following command was introduced: **bgp consistency-checker** The following command was modified: **show ip bgp vpnv4**.

CHAPTER 46

BGP Enhanced Route Refresh

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

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information for BGP Enhanced Route Refresh, on page 768.

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

Information About BGP Enhanced Route Refresh

BGP Enhanced Route Refresh Functionality

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

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

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

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

BGP Enhanced Route Refresh Timers

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

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

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

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

Syslog Messages Generated by the BGP Enhanced Route Refresh

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

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

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

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

How to Set Timers for BGP Enhanced Route Refresh

Set Timers for BGP Enhanced Route Refresh

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Router(config)# router bgp 65000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**bgp refresh stalepath-time** *seconds* **Example:** `Router(config-router)# bgp refresh stalepath-time 1200`	(Optional) Causes the router to remove stale routes from the BGP table after the timer expires, even if the router does not receive a Route-Refresh End-of-RIB message. • Valid values are from 600 to 3600, or 0. • The default is 0, meaning the stale-path timer is disabled. • The stale path timer is started when a router receives a Route-Refresh Start-of-RIB message.
Step 5	**bgp refresh max-eor-time** *seconds* **Example:** `Router(config-router)# bgp refresh max-eor-time 1200`	(Optional) Specifies that if BGP is unable to generate a Route-Refresh End-of-RIB (EOR) message, a Route-Refresh EOR is generated after the timer expires. • Valid values are from 600 to 3600, or 0. • The default is 0, meaning the max-eor timer is disabled.

Configuration Examples for BGP Enhanced Route Refresh

Example: Setting Timers for BGP Enhanced Route Refresh

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

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

Additional References

Related Documents

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

Technical Assistance

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

Feature Information for BGP Enhanced Route Refresh

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

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

Table 59: Feature Information for BGP Enhanced Route Refresh

Feature Name	Releases	Feature Information
BGP Enhanced Route Refresh	15.2(3)T	The BGP Enhanced Route Refresh feature provides a way for BGP to find route inconsistencies, and in that unlikely event, to synchronize BGP peers without a hard reset. The following commands were introduced: • **bgp refresh max-eor-time** • **bgp refresh stalepath-time**

CHAPTER 47

BGP MIB Support

The BGP MIB Support Enhancements feature introduces support in the CISCO-BGP4-MIB for new SNMP notifications.

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 MIB Support

BGP MIB Support

The Management Information Base (MIB) that supports BGP is the CISCO-BGP4-MIB. The BGP MIB Support Enhancements feature introduces support in the CISCO-BGP4-MIB for new SNMP notifications. The following sections describe the objects and notifications (traps) that are supported:

BGP FSM Transition Change Support

The cbgpRouteTable supports BGP Finite State Machine (FSM) transition state changes.

The cbgpFsmStateChange object allows you to configure SNMP notifications (traps) for all FSM transition state changes. This notification contains the following MIB objects:

- bgpPeerLastError
- bgpPeerState
- cbgpPeerLastErrorTxt
- cbgpPeerPrevState

The cbgpBackwardTransition object supports all BGP FSM transition state changes. This object is sent each time the FSM moves to either a higher or lower numbered state. This notification contains the following MIB objects:

- bgpPeerLastError
- bgpPeerState
- cbgpPeerLastErrorTxt
- cbgpPeerPrevState

The **snmp-server enable bgp traps** command allows you to enable the traps individually or together with the existing FSM backward transition and established state traps as defined in RFC 1657.

BGP Route Received Route Support

The cbgpRouteTable object supports the total number of routes received by a BGP neighbor. The following MIB object is used to query the CISCO-BGP4-MIB for routes that are learned from individual BGP peers:

- cbgpPeerAddrFamilyPrefixTable

Routes are indexed by the address-family identifier (AFI) or subaddress-family identifier (SAFI). The prefix information displayed in this table can also viewed in the output of the **show ip bgp** command.

BGP Prefix Threshold Notification Support

The cbgpPrefixMaxThresholdExceed and cbgpPrfefixMaxThresholdClear objects were introduced to allow you to poll for the total number of routes received by a BGP peer.

The cbgpPrefixMaxThresholdExceed object allows you to configure SNMP notifications to be sent when the prefix count for a BGP session has exceeded the configured value. This notification is configured on a per address family basis. The prefix threshold is configured with the **neighbor maximum-prefix** command. This notification contains the following MIB objects:

- cbgpPeerPrefixAdminLimit
- cbgpPeerPrefixThreshold

The cbgpPrfefixMaxThresholdClear object allows you to configure SNMP notifications to be sent when the prefix count drops below the clear trap limit. This notification is configured on a per address family basis. This notification contains the following objects:

- cbgpPeerPrefixAdminLimit
- cbgpPeerPrefixClearThreshold

Notifications are sent when the prefix count drops below the clear trap limit for an address family under a BGP session after the cbgpPrefixMaxThresholdExceed notification is generated. The clear trap limit is calculated by subtracting 5 percent from the maximum prefix limit value configured with the **neighbor maximum-prefix** command. This notification will not be generated if the session goes down for any other reason after the cbgpPrefixMaxThresholdExceed is generated.

VPNv4 Unicast Address Family Route Support

The cbgpRouteTable object allows you to configure SNMP GET operations for VPNv4 unicast address-family routes.

The following MIB object allows you to query for multiple BGP capabilities (for example, route refresh, multiprotocol BGP extensions, and graceful restart):

- cbgpPeerCapsTable

The following MIB object allows you to query for IPv4 and VPNv4 address family routes:

- cbgpPeerAddrFamilyTable

Each route is indexed by peer address, prefix, and prefix length. This object indexes BGP routes by the AFI and then by the SAFI. The AFI table is the primary index, and the SAFI table is the secondary index. Each BGP speaker maintains a local Routing Information Base (RIB) for each supported AFI and SAFI combination.

cbgpPeerTable Support

The cbgpPeerTable has been modified to support the enhancements described in this document. The following new table objects are supported in the CISCO-BGP-MIB.my:

- cbgpPeerLastErrorTxt
- cbgpPeerPrevState

The following table objects are not supported. The status of these objects is listed as deprecated, and these objects are not operational:

- cbgpPeerPrefixAccepted
- cbgpPeerPrefixDenied
- cbgpPeerPrefixLimit
- cbgpPeerPrefixAdvertised
- cbgpPeerPrefixSuppressed
- cbgpPeerPrefixWithdrawn

How to Enable BGP MIB Support

Enabling BGP MIB Support

SNMP notifications can be configured on the router and GET operations can be performed from an external management station only after BGP SNMP support is enabled. Perform this task on a router to configure SNMP notifications for the BGP MIB.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server enable traps bgp** [[**state-changes** [**all**] [**backward-trans**] [**limited**]] | [**threshold prefix**]] **Example:** `Device(config)# snmp-server enable traps bgp`	Enables BGP support for SNMP operations. Entering this command with no keywords or arguments enables support for all BGP events. • The **state-changes** keyword is used to enable support for FSM transition events. • The **all** keyword enables support for FSM transitions events. • The **backward-trans** keyword enables support only for backward transition state change events. • The **limited** keyword enables support for backward transition state changes and established state events. • The **threshold** and **prefix** keywords are used to enable notifications when the configured maximum prefix limit is reached on the specified peer.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode, and enters privileged EXEC mode.

Configuration Examples for BGP MIB Support

Example: Enabling BGP MIB Support

The following example enables SNMP support for all supported BGP events:

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

The following verification example shows that SNMP support for BGP is enabled by displaying any lines in the running configuration file that include “snmp-server”:

```
Device# show run | include snmp-server

snmp-server enable traps bgp
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	IP Routing: BGP Command Reference
MIB objects supported in CISCO-BGP-MIBv8.1	“Cisco-BGP-MIBv2” module in the *IP Routing: BGP Configuration Guide*
Information about SNMP and SNMP operations	SNMP Configuration Guide in the *Network Management Configuration Guide Library*

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 MIB Support

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 60: Feature Information for BGP MIB Support

Feature Name	Releases	Feature Information
BGP MIB Support Enhancements	12.0(26)S 12.2(25)S 12.3(7)T 12.2(33)SRA 12.2(22)SXH 15.0(1)SY	The BGP MIB Support Enhancements feature introduced support in the CISCO-BGP4-MIB for new SNMP notifications. The following command was introduced: **snmp-server enable traps bgp**.

CHAPTER 48

Cisco-BGP-MIBv2

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

Finding Feature Information

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

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

Information About Cisco-BGP-MIBv2

Benefits of Cisco-BGP-MIBv2

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

- MIB objects and SNMP notifications can keep track of prefix counters and capabilities related to BGP sessions.
- The following address families are supported: IPv4, IPv6, VPNv4, VPNv6.
- New traps are added to CISCO-BGP4-MIB that allow data to be returned for both IPv4 and IPv6 peers.

Cisco-BGP-MIBv2 Read-Only Objects

Cisco BGP Peer Table—cbgpPeer2Table

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

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

- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

BGP Session Capability Table—cbgpPeer2CapsTable

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

- cbgpPeer2CapCode

Peer Address Family Table—cbgpPeer2AddrFamilyTable

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

- cbgpPeer2AddrFamilyName

Prefix CounterTable—cbgpPeer2AddrFamilyPrefixTable

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

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

cbgpNotifsEnable

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

cbgpPeer2FsmStateChange

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

- cbgpPeer2LastError
- cbgpPeer2State

- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2BackwardTransition

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

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2PrefixThresholdExceeded

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

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixThreshold

cbgpPeer2PrefixThresholdClear

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

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixClearThreshold

cbgpPeer2BackwardTransNotification

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

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

Cisco-BGP-MIBv2 Trap Objects

cbgpPeer2EstablishedNotification

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

- cbgpPeer2LastError

- cbgpPeer2State

cbgpPeer2BackwardTransNotification

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

- cbgpPeer2LastError
- cbgpPeer2State

cbgpPeer2FsmStateChange

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

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2BackwardTransition

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

- cbgpPeer2LastError
- cbgpPeer2State
- cbgpPeer2LastErrorTxt
- cbgpPeer2PrevState

cbgpPeer2PrefixThresholdExceeded

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

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixThreshold

cbgpPeer2PrefixThresholdClear

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

- cbgpPeer2PrefixAdminLimit
- cbgpPeer2PrefixClearThreshold

How to Configure Cisco-BGP-MIBv2

Enabling Cisco-BGP-MIBv2 Traps

SUMMARY STEPS

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

DETAILED STEPS

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

Configuration Examples for Cisco-BGP-MIBv2

Example: Enabling Cisco-BGP-MIBv2

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

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
CISCO-BGP4-MIB support	"BGP MIB Support" module in the *IP Routing: BGP Configuration Guide*
Information about SNMP and SNMP operations	SNMP Configuration Guide

MIBs

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

Technical Assistance

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

Feature Information for Cisco-BGP-MIBv2

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

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

Table 61: Feature Information for Cisco-BGP-MIBv2

Feature Name	Releases	Feature Information
Cisco-BGP-MIBv2	15.2(3)T	The Cisco-BGP-MIBv2 feature supports CISCO-BGP-MIB, which allows MIBv2 traps to be generated. CISCO-BGP-MIB supports IPv4, IPv6, VPNv4, and VPNv6 address families. Its MIB objects and SNMP notifications can keep track of prefix counters and capabilities related to BGP sessions. The following command was modified: **snmp-server enable traps bgp**.

CHAPTER 49

BGP Graceful Shutdown

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

Finding Feature Information

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

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

Information About BGP Graceful Shutdown

Purpose and Benefits of BGP Graceful Shutdown

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

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

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

GSHUT Community

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

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

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

BGP GSHUT Enhancement

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

How to Configure BGP Graceful Shutdown

Shutting Down a BGP Link Gracefully

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	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 5000`	Configures a BGP routing process.
Step 4	**neighbor** {*ipv4-address* | *ipv6-address*} **remote-as** *number* **Example:** `Device(config-router)# neighbor 2001:db8:3::1 remote-as 5500`	Configures the autonomous system (AS) to which the neighbor belongs.
Step 5	**neighbor** {*ipv4-address* | *ipv6-address* | *peer-group-name*} **shutdown graceful** *seconds* {**community** *value* [**local-preference** *value*] | **local-preference** *value*} **Example:** `Device(config-router)# neighbor 2001:db8:3::1 shutdown graceful 600 community 1200 local-preference 300`	Configures the device to gracefully shut down the link to the specified peer in the specified number of seconds; advertises the route with the GSHUT (Graceful Shutdown) community; and advertises the route with another community or specifies a local preference value for the route, or both. • Make sure to specify an adequate amount of time for iBGP peers to converge and to choose an alternate path as the best path. • If the **graceful** keyword is used in the **neighbor shutdown** command, at least one of the two attributes (a community or local preference) must be configured. You may configure both attributes. • If the **graceful** keyword is used in the **neighbor shutdown** command, the route is advertised with the GSHUT community by default. You may also set one other community for policy routing purposes. • In this particular example, the route to the neighbor is configured to shut down in 600 seconds, is advertised with the GSHUT community and community 1200, and is configured with a local preference of 300. • The device receiving the advertisement looks at the community value(s) of the route and optionally uses the community value to apply routing policy. Filtering routes based on a community is done with the **ip community-list** command and a route map.

	Command or Action	Purpose
		• During the graceful shutdown, the **neighbor shutdown** command is not nvgened. After the timer expires, SHUTDOWN is nvgened.
Step 6	**end** **Example:** `Device(config-router)# end`	Returns to EXEC mode.
Step 7	**show ip bgp community gshut** **Example:** `Device# show ip bgp community gshut`	(Optional) Displays information about the routes that are advertised with the well-known GSHUT community.

Filtering BGP Routes Based on the GSHUT Community

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 2000`	Configures a BGP routing process.
Step 4	**neighbor** {*ipv4-address* | *ipv6-address*} **remote-as** *number* **Example:** `Device(config-router)# neighbor 2001:db8:4::1 remote-as 1000`	Configures the autonomous system (AS) to which the neighbor belongs.
Step 5	**neighbor** {*ipv4-address* | *ipv6-address*} **activate** **Example:** `Device(config-router)# neighbor 2001:db8:4::1 activate`	Activates the neighbor.
Step 6	**neighbor** {*ipv4-address* | *ipv6-address*} **send-community** **Example:** `Device(config-router)# neighbor 2001:db8:4::1 send-community`	Enables BGP community exchange with the neighbor.
Step 7	**exit** **Example:** `Device(config-router)# exit`	Exits router configuration mode.
Step 8	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map RM_GSHUT deny 10`	Configures a route map to permit or deny routes for policy routing.
Step 9	**match community** {*standard-list-number* | *expanded-list-number* | *community-list-name* [**exact**]} **Example:** `Device(config-route-map)# match community GSHUT`	Configures that the routes that match ip community-list GSHUT will be policy routed.
Step 10	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode.

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

Configuring BGP GSHUT Enhancement

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode to create or configure a BGP routing process.
Step 4	**bgp graceful-shutdown all** {**neighbors** | **vrfs**} *shutdown-time* {**community** *community-value* [**local-preference** *local-pref-value*] | **local-preference** *local-pref-value* [**community** *community-value*]} **Example:** `Device(config-router)# bgp graceful-shutdown all neighbors 180 local-preference 20 community 10`	Enables the BGP GSHUT enhancement feature on the device.
Step 5	**bgp graceful-shutdown all** {**neighbors** | **vrfs**} **activate** **Example:** `Device(config-router)# bgp graceful-shutdown all neighbors activate`	Activates graceful shutdown across all neighbors or only across VRF neighbors for BGP sessions.
Step 6	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.
Step 7	**show ip bgp** **Example:** `Device# show ip bgp neighbors 10.2.2.2 | include shutdown`	Displays entries in the BGP routing table.
Step 8	**show running-config** **Example:** `Device# show running-config | session router bgp`	Displays running configuration on the device.

Configuration Examples for BGP Graceful Shutdown

Example: Shutting Down a BGP Link Gracefully

Graceful Shutdown While Setting a Local-Preference

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

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

Graceful Shutdown While Setting an Additional Community

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

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

Graceful Shutdown while Setting an Additional Community and Local-Preference

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

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

Example: Filtering BGP Routes Based on the GSHUT Community

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

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

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

Example: BGP GSHUT Enhancement

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

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

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

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

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

Device# show ip bgp neighbors 172.16.2.1 | include shutdown

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

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

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

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

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

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

Additional References

Related Documents

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

Standards and RFCs

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

Technical Assistance

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

Feature Information for BGP Graceful Shutdown

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

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

Table 62: Feature Information for BGP Graceful Shutdown

Feature Name	Releases	Feature Information
BGP Graceful Shutdown	15.2(4)M	The BGP Graceful Shutdown feature reduces or eliminates the loss of traffic along a link being shut down for maintenance. Routers always have a valid route available during the convergence process. The following commands were modified: **ip community-list**, **neighbor shutdown**, **show ip bgp community**, and **show ip bgp vpnv4**.
BGP GSHUT Enhancement	15.4(2)T	The BGP Graceful Shutdown (GSHUT) Enhancement feature enables graceful shutdown of either all neighbors or only virtual routing and forwarding (VRF) neighbors across BGP sessions. The following command was introduced: **bgp graceful-shutdown all**.

CHAPTER 50

BGP Additional Paths

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

Finding Feature Information

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

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

Information About BGP Additional Paths

Problem That Additional Paths Can Solve

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

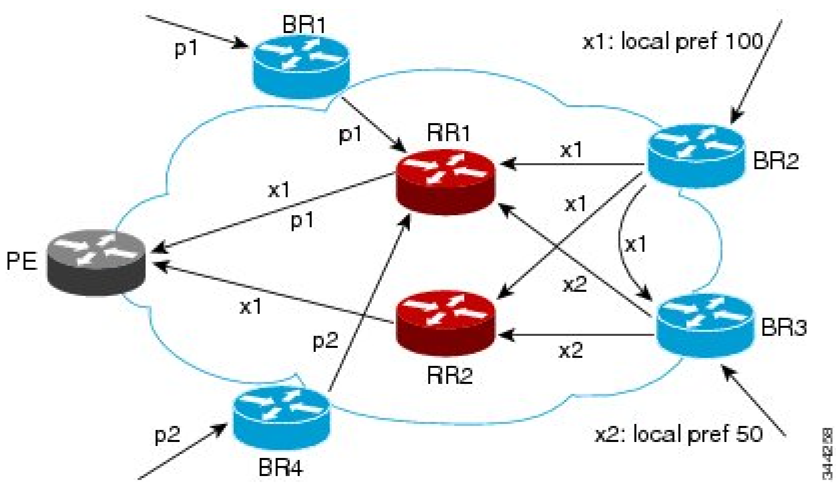
Path hiding can prevent efficient use of BGP multipath, prevent hitless planned maintenance, and can lead to MED oscillations and suboptimal hot-potato routing. Upon nexthop failures, path hiding also inhibits fast and local recovery because the network has to wait for BGP control plane convergence to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

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

Path-Hiding Scenario

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

Figure 70: RR Hiding an Additional Path

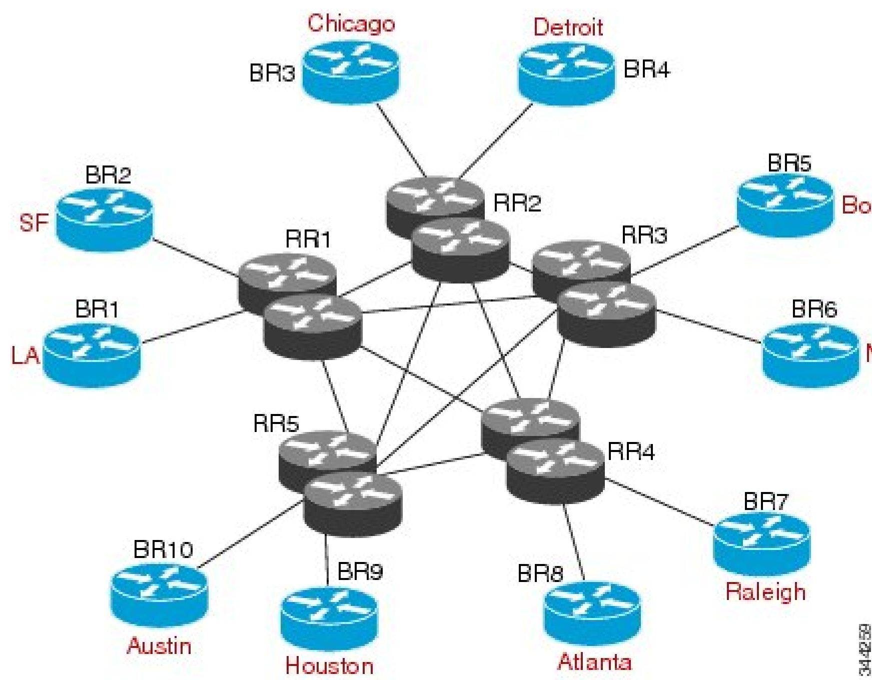


In the figure above, we also see prefix x with path x1 being advertised from BR2 to BR3 (which has path x2) with local preference 100. BR3 also has path x2, but due to routing policy, BR3 will advertise to the RRs x1 (not shown) instead of x2, and x2 will be suppressed. A user could enable the advertisement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

Suboptimal Hot-Potato Routing Scenario

In order to minimize internal transport costs, transit ISPs try to forward packets to the closest exit point (according to Interior Gateway Protocol [IGP] cost). This behavior is known as hot-potato routing. In the distributed RR cluster model of the figure below, assume traffic coming from LA must go to Mexico. All links have the same IGP cost. If there are two exit points toward Mexico—one toward Austin and one toward Atlanta—the border router will try to send traffic to Austin based on the lower IGP cost from LA toward Austin than toward Atlanta. In a centralized RR model where the central RR resides where RR3 is (and RR1, RR2, RR4, and RR5 do not exist), the closest exit point toward Mexico, as seen from RR3, might be Atlanta. Sending the traffic from LA toward Atlanta results in suboptimal hot-potato routing, which is not desirable.

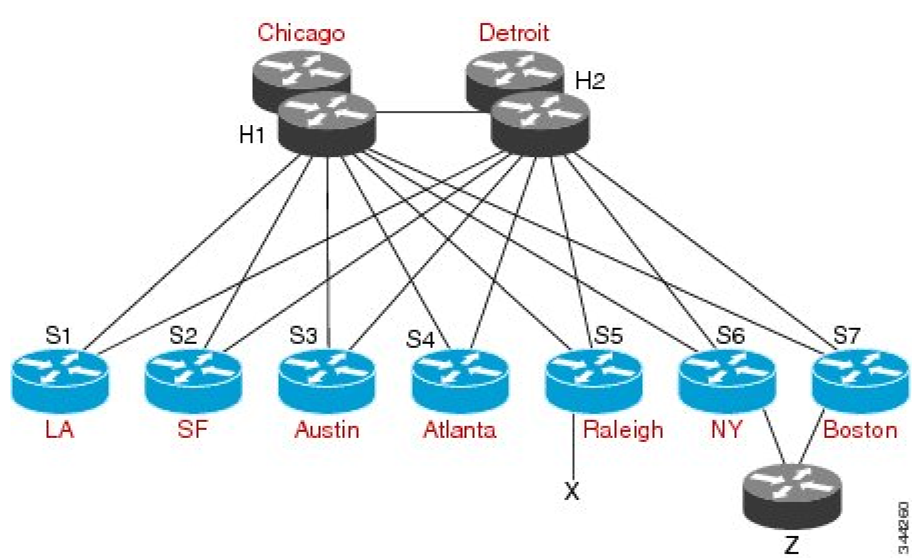
Figure 71: Distributed RR Cluster



DMVPN Scenario

In Dynamic Multipoint Virtual Private Network (DMVPN) deployments, BGP is being used for scaling. In the figure below, Z is connected to both spokes S6 (NY) and S7 (Boston). The S7 links to the hubs have lower IGP costs than the S6 links to the hubs. There are physical links not shown that connect S5 to S6 and S6 to S7, with IGP costs lower than those to the hubs. Spokes S6 and S7 will send an update to both hubs H1 (Chicago) and H2 (Detroit). The RR hubs will then select the best path based on their lower IGP cost, which might be S7. The spoke S5 (Raleigh) will receive two updates from the RRs for Z with S7 being the next hop, even though, in this scenario, it might be preferable to pick S6 (NY) as the next hop.

Figure 72: DMVPN Deployment



Benefits of BGP Additional Paths

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

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

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

BGP Additional Paths Functionality

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

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

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

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

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

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

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

Additional Path Selection

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

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

Definition of the group-best Selection

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

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

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

Advertise a Subset of the Paths Selected

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

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

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

How to Configure BGP Additional Paths

Configuring Additional Paths per Address Family

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**address-family ipv4** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv4`	Enters address family configuration mode. • The following address families are supported: IPv4 unicast, IPv4 multicast, IPv4 unicast + label, IPv6 unicast, IPv6 multicast, and IPv6 multicast + label.
Step 5	**bgp additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-af)# bgp additional-paths send receive`	Enables BGP additional paths to be sent only, received only, or sent and received, after negotiation with the neighbor is completed. • This example enables additional paths to be sent and received.

	Command or Action	Purpose
Step 6	**bgp additional-paths select group-best** **Example:** `Device(config-router-af)# bgp additional-paths select group-best`	(Optional) Calculates the group-best for prefixes during bestpath calculation.
Step 7	**bgp additional-paths select best** *number* **Example:** `Device(config-router-af)# bgp additional-paths select best 3`	(Optional) Calculates the specified number of best paths, including the advertisement of the bestpath. • The value of *number* can be 2 or 3.
Step 8	**bgp additional-paths select all** **Example:** `Device(config-router-af)# bgp additional-paths select all`	(Optional) Specifies that all paths with unique next hops are eligible for selection.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name* } **advertise additional-paths** [**best** *number*] [**group-best**] [**all**] **Example:** `Device(config-router-af)# neighbor 192.168.0.1 advertise additional-paths best 3 group-best all`	Specifies which selection methods control the additional paths that are advertised to the neighbor.
Step 10	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits to privileged EXEC mode.

Configuring Additional Paths per Neighbor

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Configures a BGP routing process and enters router configuration mode. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**address-family ipv4** [**unicast** | **multicast**] **Example:** `Device(config-router)# address-family ipv4 unicast`	Enters address family configuration mode. • The following address families are supported: IPv4 unicast, IPv4 multicast, IPv4 unicast + label, IPv6 unicast, IPv6 multicast, and IPv6 multicast + label.
Step 5	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-af)# neighbor 192.168.1.2 additional-paths send receive`	Enables the neighbor to send or receive additional paths after negotiation is completed. • This example enables the neighbor to send and receive additional paths. • Note that this command overrides any send or receive capability that might have been configured at the address-family level.
Step 6	**bgp additional-paths select group-best** **Example:** `Device(config-router-af)# bgp additional-paths select group-best`	(Optional) Calculates the group-best for prefixes during bestpath calculation.
Step 7	**bgp additional-paths select best** *number* **Example:** `Device(config-router-af)# bgp additional-paths select best 3`	(Optional) Calculates the specified number of best paths, including the selection of the bestpath. • The value of *number* can be 2 or 3.
Step 8	**bgp additional-paths select all** **Example:**	(Optional) Specifies that all paths with unique next hops are eligible for selection.

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

Configuring Additional Paths Using a Peer Policy Template

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Configures the IPv4 address family.
Step 5	**bgp additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router)# bgp additional-paths send receive`	Enables BGP additional paths to be sent only, received only, or sent and received for the peers in the address family.
Step 6	**bgp additional-paths select** [**best** *number*] [**group-best**] [**all**] **Example:** `Device(config-router)# bgp additional-paths select best 3 group-best all`	Causes the system to calculate BGP additional paths that can be candidates for advertisement in addition to a bestpath.
Step 7	**template peer-policy** *policy-template-name* **Example:** `Device(config-router)# template peer-policy rr-client-pt1`	Enters policy-template configuration mode and creates a peer policy template.
Step 8	**additional-paths** {**send** [**receive**] | **receive**} **Example:** `Device(config-router-ptmp)# additional-paths send receive`	Enables BGP additional paths to be sent only, received only, or sent and received for the peers covered by the peer policy template.
Step 9	**advertise additional-paths** [**best** *number*] [**group-best**] [**all**] **Example:** `Device(config-router-ptmp)# advertise additional-paths best 3 group-best all`	Specifies the selection methods that control which additional paths are advertised for the peers covered by the peer policy template.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Device(config-router-ptmp)# exit`	Exits policy-template configuration mode and returns to router configuration mode.
Step 11	**address-family ipv4 unicast** **Example:** `Device(config-router)# address-family ipv4 unicast`	Configures the IPv4 address family.
Step 12	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 remote-as 45000`	Adds an entry to the BGP neighbor table.
Step 13	**neighbor** *ip-address* **inherit peer-policy** *policy-template-name* **Example:** `Device(config-router-af)# neighbor 192.168.1.1 inherit peer-policy rr-client-pt1`	Sends a peer policy template to a neighbor so that the neighbor can inherit the configuration.
Step 14	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Filtering and Setting Actions for Additional Paths

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

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

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

SUMMARY STEPS

1. **enable**

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map additional_path1 permit 10`	Creates a route map.
Step 4	**match additional-paths advertise-set** [**best** *number*] [**best-range** *start-range end-range*] [**group-best**] [**all**] **Example:** `Device(config-route-map)# match additional-paths advertise-set best 3`	Matches on any path that is tagged with the specified path selection policy. • You must specify at least one selection method; you can specify more than one selection method in the command. • Specifying **best** *number* is incompatible with specifying **best-range**. • Specifying **best 1** will match only the bestpath. • Specifying **best-range 1 1** will match only the bestpath. • Only one **match additional-paths advertise-set** command is allowed per route map. A subsequent **match additional-paths advertise-set** command will overwrite the previous command.
Step 5	**set metric** *metric-value* **Example:** `Device(config-route-map)# set metric 500`	Sets the metric of the additional paths that pass the match criteria. • Note that other **set** commands can be used to take action on the paths that pass the route map. This example happens to use the **set metric** command.

What to do next

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

Displaying Additional Path Information

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip bgp neighbors** [*ip-address*] **Example:** `Device# show ip bgp neighbors 192.168.1.1`	Displays the capabilities of the neighbor to send and receive additional paths.
Step 3	**show ip bgp** [*network*] **Example:** `Device# show ip bgp 192.168.0.0`	Displays the additional path selections and path ID for the network.

Disabling Additional Paths per Neighbor

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 2001:DB8::1 additional-paths disable`	
Step 7	**end** **Example:** `Device(config-router-af)# end`	(Optional) Exits to privileged EXEC mode.

Configuration Examples for BGP Additional Paths

Example: BGP Additional Path Send and Receive Capabilities

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

R1

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

R2

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

Example: BGP Additional Paths

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

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

- In the IPv6 multicast address family, all paths are candidates to be advertised and will be advertised toward neighbor 2001:DB8::1037.

```
router bgp 1
 neighbor 192.168.101.15 remote-as 1
 neighbor 192.168.101.25 remote-as 1
 neighbor 2001:DB8::1045 remote-as 1
 neighbor 2001:DB8::1037 remote-as 1
!
 address-family ipv4 unicast
  bgp additional-paths send receive
  bgp additional-paths select all best 3 group-best
  neighbor 192.168.101.15 activate
  neighbor 192.168.101.15 route-map add_path1 out
  neighbor 192.168.101.15 advertise additional-paths best 2
 exit-address-family
!
address-family ipv4 multicast
  bgp additional-paths send receive
  bgp additional-paths select all best 3 group-best
  neighbor 192.168.101.25 activate
  neighbor 192.168.101.25 route-map add_path2 out
  neighbor 192.168.101.25 advertise additional-paths best 3
 exit-address-family
!
 address-family ipv6 unicast
  bgp additional-paths send receive
  bgp additional-paths select group-best
  neighbor 2001:DB8::1045 activate
  neighbor 2001:DB8::1045 route-map add_path3 out
  neighbor 2001:DB8::1045 advertise additional-paths all group-best
 exit-address-family
!
 address-family ipv6 multicast
  bgp additional-paths send receive
  bgp additional-paths select all
  neighbor 2001:DB8::1037 activate
  neighbor 2001:DB8::1037 route-map add_path4 out
  neighbor 2001:DB8::1037 advertise additional-paths all
 exit-address-family
!
route-map add_path1 permit 10
 match additional-paths advertise-set best 2
 set metric 780
route-map add_path1 permit 20
!
route-map add_path2 permit 10
 match additional-paths advertise-set best 3
 set metric 640
!
route-map add_path3 permit 10
 match additional-paths advertise-set group-best
 set metric 825
!
```

Example: Neighbor Capabilities Override Address Family Capabilities

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

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

Example: BGP Additional Paths Using a Peer Policy Template

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

Additional References

Related Documents

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

Standards and RFCs

Standard/RFC	Title
RFC 3107	*Carrying Label Information in BGP-4*
RFC 4271	*A Border Gateway Protocl (BGP-4)*
RFC 4760	*Multiprotocol Extensions for BGP-4*

Technical Assistance

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

Feature Information for BGP Additional Paths

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

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

Table 63: Feature Information for BGP Additional Paths

Feature Name	Releases	Feature Information
BGP Additional Paths	15.3(1)T	The BGP Additional Paths feature allows the advertisement of multiple paths for the same prefix without the new paths implicitly replacing any previous paths. The following commands were introduced: • **additional-paths** • **advertise additional-paths** • **bgp additional-paths** • **bgp additional-paths select** • **match additional-paths advertise-set** • **neighbor additional-paths** • **neighbor advertise additional-paths** The following commands were modified: • **show ip bgp** • **show ip bgp neighbors**

CHAPTER 51

BGP Attribute Filter and Enhanced Attribute Error Handling

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

Finding Feature Information

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

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

Information About BGP Attribute Filtering

BGP Attribute Filter and Enhanced Attribute Error Handling

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

- The feature allows you to treat-as-withdraw an Update coming from a specified neighbor if the Update contains a specified attribute type. When an Update is treat-as-withdraw, the prefixes in the Update are removed from the BGP routing table (if they existed in the routing table).

- The feature also allows you to drop specified path attributes from an Update, and then the system processes the rest of the Update as usual.

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

The features are implemented in the following order:

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

Details About Specifying Attributes as Treat-as-Withdraw

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

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

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

Details About Specifying Attributes as Discard

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

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

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

Details About Enhanced Attribute Error Handling

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

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

How to Filter BGP Path Attributes

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

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
Step 5	Repeat Step 4 to configure other attributes not in a range or to configure a different neighbor.	
Step 6	**end** **Example:** `Device(config-router)# end`	Exits to privileged EXEC mode.

Discarding Specific Path Attributes from an Update Message

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 6500`	Configures a BGP routing process and enters router configuration mode.
Step 4	**neighbor** {*ip-address* | *ipv6-address*} **path-attribute discard** {*attribute-value* | **range** *start-value end-value*} **in** **Example:**	Drops specified path attributes from Update messages from the specified neighbor.

	Command or Action	Purpose
	`Device(config-router)# neighbor 2001:DB8:1::1 path-attribute discard 128 in`	
Step 5	Repeat Step 4 to configure other attributes not in a range or to configure a different neighbor. **Example:**	
Step 6	**end** **Example:** `Device(config-router)# end`	Exits to privileged EXEC mode.

Displaying Withdrawn or Discarded Path Attributes

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
Step 4	**show ip bgp path-attribute discard** **Example:** `Device# show ip bgp path-attribute discard`	(Optional) Displays all prefixes for which an attribute has been discarded.
Step 5	**show ip bgp vpnv4 all** *prefix* **Example:** `Device# show ip bgp vpnv4 all 192.168.1.0`	(Optional) Displays the unknown attributes and discarded attributes associated with a prefix.
Step 6	**show ip bgp neighbors** *prefix* **Example:** `Device# show ip bgp neighbors 192.168.1.0`	(Optional) Displays the configured discard and treat-as-withdraw attributes associated with a prefix.

Configuration Examples for BGP Attribute Filter

Examples: Withdraw Updates Based on Path Attribute

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

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

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

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

Examples: Discard Path Attributes from Updates

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

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

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

```
router bgp 65600
 neighbor 2001:DB8:1::1 path-attribute discard 17 255 in
```

Additional References

Related Documents

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

Standards and RFCs

Standard/RFC	Title
draft-ietf-idr-error-handling	Revised Error Handling for BGP Updates from External Neighbors

Technical Assistance

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

Feature Information for BGP Attribute Filter and Enhanced Attribute Error Handling

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

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

Table 64: Feature Information for BGP Attribute Filter and Enhanced Attribute Error Handling

Feature Name	Releases	Feature Information
BGP Attribute Filter and Enhanced Attribute Error Handling	15.3(1)T	The BGP Attribute Filter allows you to “treat-as-withdraw” updates that contain specific path attributes. The prefixes contained in the update are removed from the routing table. The feature also allows you to remove specific path attributes from incoming updates. Both behaviors provide an increased measure of security. The BGP Enhanced Attribute Error Handling feature prevents peer sessions from flapping due to errors from any malformed update, thereby saving resources. The following commands were introduced: **bgp enhanced-error**, **neighbor path-attribute discard**, **neighbor path-attribute treat-as-withdraw**, **show ip bgp path-attribute discard**, and **show ip bgp path-attribute unknown**. The following commands were modified: **show ip bgp**, **show ip bgp neighbor**, and **show ip bgp vpnv4 all**.

CHAPTER 52

BGP MVPN Source-AS Extended Community Filtering

The BGP MVPN Source-AS Extended Community Filtering feature enables the provider edge (PE) device to suppress attaching the multicast VPN (MVPN)-related extended communities to routes learned from a customer edge (CE) device or redistributed in a virtual routing and forwarding (VRF) instance for a specified neighbor.

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 MVPN Source-AS Extended Community Filtering

Overview of BGP MVPN Source-AS Extended Community Filtering

VPN routes carry special extended communities (source autonomous system [AS] extended community and virtual routing and forwarding [VRF] route import extended community) to support multicast VPN (MVPN). Legacy provider edge (PE) devices interpret the source AS extended community as old style multicast distribution tree (MDT). You can attach the extended communities when the prefix is created. After the BGP

MVPN Source-AS Extended Community Filtering feature is enabled, this allows the PE device to suppress these extended communities. You can use this functionality to suppress extended communities from being sent for Subsequent Address Family Identifier (SAFI) 128 routes and instead use SAFI 129. Devices with SAFI 129 must be able to identify the source AS extended community correctly.

How to Configure BGP MVPN Source-AS Extended Community Filtering

Configuring BGP MVPN Source-AS Extended Community Filtering

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address-family ipv4 vrf** *vrf-name*
5. **unicast-reachability** [**source-as** | **vrf-route-import**] [**disable**]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family ipv4 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 **vrf** keyword and *vrf-name* argument to specify the name of the VRF instance to associate with subsequent IPv4 address family configuration mode commands.

	Command or Action	Purpose
Step 5	**unicast-reachability** [**source-as** | **vrf-route-import**] [**disable**] **Example:** `Device(config-router-af)# unicast-reachability source-as disable`	Disables advertising extended communities for non-MVPN profiles.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Configuration Examples for BGP MVPN Source-AS Extended Community Filtering

Example: Configuring BGP MVPN Source-AS Extended Community Filtering

The following example configures BGP MVPN source-AS extended community filtering:

```
Device# configure terminal
Device(config)# router bgp 45000
Device(config)# address-family ipv4 vrf vpn1
Device(config-router-af)# unicast-reachability source-as disable
Device(config-router-af)# exit
```

The following example shows summary output for the **show ip bgp vpnv4 vrf vpn1** command.

```
Device# show ip bgp vpnv4 vrf vpn1

BGP routing table entry for 10:10:1.1.1.1/32, version 25
Paths: (2 available, best #2, table red)
Multipath: eiBGP
  Advertised to update-groups:
     1
  Refresh Epoch 1
  Local, imported path from 10:11:1.1.1.1/32 (global)
    1.1.1.2 (metric 11) (via default) from 1.1.1.5 (1.1.1.5)
      Origin incomplete, metric 11, localpref 100, valid, internal
      Extended Community: RT:1:1 OSPF DOMAIN ID:0x0005:0x000000C80200
        MVPN AS:55:0.0.0.0 MVPN VRF:1.1.1.2:2 OSPF RT:0.0.0.0:2:0
        OSPF ROUTER ID:10.10.20.2:0
      Originator: 1.1.1.2, Cluster list: 1.1.1.5
      Connector Attribute: count=1
       type 1 len 12 value 10:11:1.1.1.2
      mpls labels in/out 20/21
      rx pathid: 0, tx pathid: 0
  Refresh Epoch 1
  Local
    10.10.10.100 (via vrf red) from 0.0.0.0 (1.1.1.1)
      Origin incomplete, metric 11, localpref 100, weight 32768, valid, sourced, best
      Extended Community: RT:1:1 OSPF DOMAIN ID:0x0005:0x000000C80200
        MVPN VRF:1.1.1.1:1 OSPF RT:0.0.0.0:2:0 OSPF ROUTER ID:10.10.10.1:0
```

```
mpls labels in/out 20/nolabel
rx pathid: 0, tx pathid: 0x0
```

Additional References for BGP MVPN Source-AS Extended Community Filtering

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
BGP concepts and tasks	*IP Routing: BGP Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP MVPN Source-AS Extended Community 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 65: Feature Information for BGP MVPN Source-AS Extended Community Filtering

Feature Name	Releases	Feature Information
BGP MVPN Source-AS Extended Community Filtering	15.5(1)T	The BGP MVPN Source-AS Extended Community Filtering feature enables the provider edge (PE) device to suppress attaching the multicast VPN (MVPN)-related extended communities to routes learned from a customer edge (CE) device or redistributed in a virtual routing and forwarding (VRF) instance for a specified neighbor. The following command was introduced or modified: **unicast-reachability**.

CHAPTER 53

BGP-MVPN SAFI 129 IPv6

Subsequent Address Family Identifier (SAFI) 129, known as VPN Multicast SAFI, provides the capability to support multicast routing in the service provider's core IPv6 network.

Border Gateway Protocol (BGP) Multicast Virtual Private Network (MVPN) provides a means for service providers to use different encapsulation methods (generic routing encapsulation [GRE], Multicast Label Distribution Protocol [MLDP], and ingress replication) for forwarding MVPN multicast data traffic in the service provider network.

The BGP-MVPN SAFI 129 IPv6 feature is required to support BGP-based MVPNs.

Finding Feature Information

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

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

Prerequisites for BGP-MVPN SAFI 129 IPv6

- Before you configure a SAFI 129 IPv6-related address family, the **ipv6 unicast-routing** command must be configured on the device.
- To create a multicast IPv6 VRF address family under BGP, IPv6 must first be activated on the VRF itself.

Note There is no separate multicast configuration on the VRF. Configuring the **address-family ipv6** command on the VRF will enable both unicast and multicast topologies.

- If you want prefixes to be installed into the Routing Information Base (RIB), you must configure the **pim** command on a VRF interface.

Information About BGP-MVPN SAFI 129 IPv6

Overview of BGP-MVPN SAFI 129 IPv6

MVPN utilizes the existing VPN infrastructure to allow multicast traffic to pass through the provider space. Information derived from VPN routes is one of the components needed to set up tunnels within the core. Currently, multicast traffic will derive this information from the unicast VPNv6 tables, which forces multicast traffic to be dependent on unicast topologies.

For scenarios in which multicast and unicast traffic would be better suited with separate topologies, the customer edge (CE) router may advertise a special set of routes to be used exclusively for multicast VPNs. Multicast routes learned from the CE router can be propagated to remote provider edge (PE) routers via SAFI 129. Multicast routes learned from the CE router or multicast VPN routes learned from remote PE routers can now be installed directly into the multicast RIB, instead of using replicated routes from the unicast RIB. Maintaining separate routes and entries for unicast and multicast allows you to create differing topologies for each service within the core.

How to Configure BGP-MVPN SAFI 129 IPv6

Configuring BGP-MVPN SAFI 129 IPv6

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vrf definition** *vrf1*
4. **rd** *route-distinguisher*
5. **route-target export** *route-target-ext-community*
6. **route-target import** *route-target-ext-community*
7. **address-family ipv6**
8. **mdt default** *group-address*
9. **exit**
10. **exit**
11. **router bgp** *autonomous-system-number*
12. **address-family vpnv6 multicast**

13. **neighbor** *peer-group-name* **send-community extended**
14. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate**
15. **address-family ipv6 multicast vrf** *vrf-name*
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	**vrf definition** *vrf1* **Example:** `Device(config)# vrf definition vrf1`	Defines a VRF instance and enters VRF configuration mode.
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 1:1`	Specifies a route distinguisher (RD) for a VRF instance.
Step 5	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 1:1`	Creates a route target export extended community for a VRF instance.
Step 6	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 1:1`	Creates a route target import extended community for a VRF instance.
Step 7	**address-family ipv6** **Example:** `Device(config-vrf)# address-family ipv6`	Configures a routing session using IPv6 address prefixes and enters address family configuration mode.
Step 8	**mdt default** *group-address* **Example:** `Device(config-vrf-af)# mdt default 239.0.0.1`	Configures a default multicast distribution tree (MDT) group for a VRF instance.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-vrf-af)# exit`	Exits address family configuration mode and enters VRF configuration mode.
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 50000`	Configures a BGP routing process and enters router configuration mode.
Step 12	**address-family vpnv6 multicast** **Example:** `Device(config-router)# address-family vpnv6 multicast`	Configures a routing session using VPN Version 6 multicast address prefixes and enters address family configuration mode.
Step 13	**neighbor** *peer-group-name* **send-community extended** **Example:** `Device(config-router-af)# neighbor client1 send-community extended`	Specifies that a communities attribute should be sent to a BGP neighbor.
Step 14	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address* **%**} **activate** **Example:** `Router(config-router-af)# neighbor 2001:DB8:0:CC00::1 % activate`	Enables the neighbor to exchange prefixes for the specified family type with the neighbor and the local router.
Step 15	**address-family ipv6 multicast vrf** *vrf-name* **Example:** `Device(config-router-af)# address-family ipv6 multicast vrf vrf1`	Configures a routing session using IPv6 multicast address prefixes for a VRF instance.
Step 16	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP-MVPN SAFI 129 IPv6

Example: Configuring BGP-MVPN SAFI 129 IPv6

The example below shows the configuration for a PE router:

```
hostname PE1
!
!
vrf definition blue
 rd 55:1111
 route-target export 55:1111
 route-target import 55:1111
 !
 address-family ipv6
  mdt default 232.1.1.1
  mdt data 232.1.200.0 0.0.0.0
 exit-address-family
!
!ip multicast-routing
ip multicast-routing vrf blue
ip cef
!
ipv6 unicast-routing
ipv6 multicast-routing
ipv6 multicast-routing vrf blue
ipv6 cef
!
!interface Loopback0
 ip address 205.1.0.1 255.255.255.255
 ip pim sparse-dense-mode
 ipv6 address FE80::205:1:1 link-local
 ipv6 address 205::1:1:1/64
 ipv6 enable
!
interface Ethernet0/0
 ! interface connect to the core vpn
 bandwidth 1000
 ip address 30.3.0.1 255.255.255.0
 ip pim sparse-dense-mode
 delay 100
 ipv6 address FE80::70:1:1 link-local
 ipv6 address 70::1:1:1/64
 ipv6 enable
 mpls ip
!
interface Ethernet1/1
 ! interface connect to CE (vrf interface)
 bandwidth 1000
 vrf forwarding blue
 ip address 10.1.0.1 255.255.255.0
 ip pim sparse-dense-mode
 delay 100
 ipv6 address FE80::20:1:1 link-local
 ipv6 address 20::1:1:1/64
 ipv6 enable
!
router ospf 200
```

```
 redistribute connected subnets
 redistribute bgp 55 metric 10
 passive-interface Loopback0
 network 30.3.0.0 0.0.255.255 area 1
!
router bgp 55
 bgp log-neighbor-changes
 no bgp default route-target filter
 ! neighbor to another PE in core
 neighbor 205.3.0.3 remote-as 55
 neighbor 205.3.0.3 update-source Loopback0
 !
 address-family ipv4 mdt
  ! neighbor to another PE in core
  neighbor 205.3.0.3 activate
  neighbor 205.3.0.3 send-community extended
 exit-address-family
 !
 address-family vpnv6
  ! neighbor to another PE in core
  neighbor 205.3.0.3 activate
  neighbor 205.3.0.3 send-community extended
 exit-address-family
 !
 address-family vpnv6 multicast
  ! neighbor to another PE in core
  ! this address-family is added to enable
  ! safi129 between two PEs
  neighbor 205.3.0.3 activate
  neighbor 205.3.0.3 send-community extended
 exit-address-family
 !
 address-family ipv6 vrf blue
  ! neighbor to CE1 in vrf
  redistribute connected
  redistribute static
  neighbor FE80::20:1:6%Ethernet1/1 remote-as 56
  neighbor FE80::20:1:6%Ethernet1/1 activate
 exit-address-family
 !
 address-family ipv6 multicast vrf blue
  !  neighbor to CE1 in vrf
  ! this address-family is added to enable
  ! safi2 on PE-CE
  redistribute connected
  redistribute static
  neighbor FE80::20:1:6%Ethernet1/1 remote-as 56
  neighbor FE80::20:1:6%Ethernet1/1 activate
 exit-address-family
!
ipv6 pim vrf blue rp-address 201::1:1:7 blue_bidir_acl bidir
ipv6 pim vrf blue rp-address 202::1:1:6 blue_sparse_acl
!
ipv6 access-list black_bidir_acl
 permit ipv6 any FF06::/64
!
ipv6 access-list black_sparse_acl
 permit ipv6 any FF04::/64
!
ipv6 access-list blue_bidir_acl
 permit ipv6 any FF05::/64
!
ipv6 access-list blue_sparse_acl
 permit ipv6 any FF03::/64
```

```
!
end
```

The example below shows the configuration for a CE router:

```
hostname CE1
!
ip multicast-routing
ip cef
ipv6 unicast-routing
ipv6 multicast-routing
ipv6 multicast rpf use-bgp
ipv6 cef
!
interface Ethernet1/1
 bandwidth 1000
 ip address 10.1.0.6 255.255.255.0
 no ip redirects
 no ip proxy-arp
 ip pim sparse-dense-mode
 delay 100
 ipv6 address FE80::20:1:6 link-local
 ipv6 address 20::1:1:6/64
 ipv6 enable
 no keepalive
!
router bgp 56
 bgp log-neighbor-changes
 neighbor FE80::20:1:1%Ethernet1/1 remote-as 55
 !
 address-family ipv6
  redistribute connected
  redistribute static
  neighbor FE80::20:1:1%Ethernet1/1 activate
 exit-address-family
 !
 address-family ipv6 multicast
  redistribute connected
  redistribute static
  neighbor FE80::20:1:1%Ethernet1/1 activate
 exit-address-family
!
ipv6 pim rp-address 201::1:1:7 blue_bidir_acl bidir
ipv6 pim rp-address 202::1:1:6 blue_sparse_acl
!
ipv6 access-list blue_bidir_acl
 permit ipv6 any FF05::/64
!
ipv6 access-list blue_sparse_acl
 permit ipv6 any FF03::/64
!
end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
BGP commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Routing: BGP Command Reference*

Standards and RFCs

Standard/RFC	Title
MDT SAFI	*Subsequent Address Family Identifiers (SAFI) Parameters*
RFC 2547	*BGP/MPLS VPNs*

Technical Assistance

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

Feature Information for BGP-MVPN SAFI 129 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 66: Feature Information for BGP—MVPN SAFI 129 IPv6

Feature Name	Releases	Feature Information
BGP—MVPN SAFI 129 IPv6	15.2(4)S Cisco IOS XE Release 3.7S 15.3(1)T	SAFI 129 , known as VPN Multicast SAFI, provides the capability to support multicast routing in the service provider's core IPv6 network. The following commands were introduced or modified: **address-family ipv6**, **address-family vpnv6**, and **show bgp vpnv6 multicast**.

CHAPTER 54

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

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

Finding Feature Information

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

- **match as-path**
- **match community** [**exact-match**]
- **match extcommunity**
- **match ip address** [**prefix-list**]
- **match ip next-hop**
- **match ip route-source**
- **match ipv6 address** [**prefix-list**]
- **match ipv6 route-source**
- **match ipv6 next-hop**

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

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

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

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

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

SUMMARY STEPS

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

13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vrf definition** *vrf-name* **Example:** `Device(config)# vrf definition vpn1`	Creates a VRF routing table and specifies the VRF name (or tag).
Step 4	**rd** *route-distinguisher* **Example:** `Device(config-vrf)# rd 100:100`	Creates routing and forwarding tables for the VRF instance. • There are two formats for configuring the argument. It can be configured in the *as-number*:*network number* (*ASN*:*nn*) format, as shown in the example, or it can be configured in the *IP address*:*network number* format (*IP-address*:*nn*).
Step 5	**address-family** {**ipv4** | **ipv6**} **Example:** `Device(config-vrf)# address-family ipv4`	Configures the IPv4 or IPv6 address family.
Step 6	**export** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*prefix-limit*] **map** *map-name* **Example:** `Device(config-vrf-af)# export ipv4 unicast 500 map UNICAST`	Exports IPv4 or IPv6 prefixes from the VRF table to the global routing table, filtered by the specified route map. • Specify **ipv4** or **ipv6**, which you specified in Step 5. This example exports IPv4 unicast prefixes. • Based on this example, no more than 500 prefixes will be exported. • The prefixes exported are those that pass the route map.
Step 7	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf-af)# route-target import 100:100`	Creates a route-target extended community for a VRF instance. • For information about route-target import or export, see the *MPLS: Layer 3 VPNs Configuration Guide*.

	Command or Action	Purpose
Step 8	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf-af)# route-target export 100:100`	Creates a route-target extended community for a VRF instance.
Step 9	**exit** **Example:** `Device(config-vrf-af)# exit`	Exits address family configuration mode and enters global configuration mode.
Step 10	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 11	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map UNICAST permit 10`	Enables policy routing. • The example creates a route map named UNICAST.
Step 12	**match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]} **Example:** `Device(config-route-map)# match ip address 50`	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, and performs policy routing on matched packets. • Both IP access lists and IP prefix lists are supported. • The example configures the route map to use standard access list 50 to define match criteria. • Define the access list (not shown in this task); for example, access-list 50 permit 192.168.1.0 255.255.255.0.
Step 13	**end** **Example:** `Device(config-route-map)# end`	Exits route-map configuration mode and returns to privileged EXEC mode.

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

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

Note

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

SUMMARY STEPS

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

DETAILED STEPS

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

	Command or Action	Purpose
		be configured in the *IP-address*:*network number* format (*IP-address*:*nn*).
Step 5	**export ipv4** {**unicast** | **multicast**} [*prefix-limit*] **map** *map-tag* **Example:** `Device(config-vrf)# export ipv4 unicast 500 map UNICAST`	Exports IPv4 prefixes from the VRF table to the global routing table, filtered by the specified route map. • Unicast or multicast prefixes are specified. • By default, up to 1000 prefixes can be exported. The *prefix-limit* argument is used to specify a limit from 1 to 2,147,483,647 prefixes. • The example creates an export map that will export up to 500 unicast prefixes that pass through the route map named UNICAST.
Step 6	**route-target import** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target import 100:100`	Creates a route-target extended community for a VRF instance. • For information about route-target import or export, see the *MPLS: Layer 3 VPNs Configuration Guide*.
Step 7	**route-target export** *route-target-ext-community* **Example:** `Device(config-vrf)# route-target export 100:100`	Creates a route-target extended community for a VRF instance.
Step 8	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 9	**route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] **Example:** `Device(config)# route-map UNICAST permit 10`	Defines the conditions for redistributing routes from one routing protocol into another, or enables policy routing. • The route map name must match the route map specified in Step 5. • The example creates a route map named UNICAST.
Step 10	**match ip address** {*acl-number* [*acl-number* | *acl-name*] | *acl-name* [*acl-name* | *acl-number*] | **prefix-list** *prefix-list-name* [*prefix-list-name*]} **Example:** `Device(config-route-map)# match ip address 50`	Distributes any routes that have a destination network number address that is permitted by a standard or extended access list, and performs policy routing on matched packets. • Both IP access lists and IP prefix lists are supported. • The example configures the route map to use standard access list 50 to define match criteria.

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

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

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

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

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

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

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

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

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

Additional References

Related Documents

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

Technical Assistance

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

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

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

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

Table 67: Feature Information for BGP Support for IP Prefix Export from a VRF Table into the Global Table

Feature Name	Releases	Feature Information
BGP Support for IP Prefix Export from a VRF Table into the Global Table	15.3(1)T	This feature allows a network administrator to export IP prefixes from a VRF routing table into the global routing table. The following command was introduced: **export map (VRF table to global table)**. The following commands were modified: **debug ip bgp import** and **show ip bgp**.

CHAPTER 55

BGP-Multiple Cluster IDs

The BGP—Multiple Cluster IDs feature allows an iBGP neighbor (usually a route reflector) to have multiple cluster IDs: a global cluster ID and additional cluster IDs that are assigned to clients (neighbors). Prior to the introduction of this feature, a device could have a single, global cluster ID.

When a network administrator configures per-neighbor cluster IDs:

- The loop prevention mechanism based on a CLUSTER_LIST is automatically modified to take into account multiple cluster IDs.
- A network administrator can disable client-to-client route reflection based on cluster ID.

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-Multiple Cluster IDs

Benefit of Multiple Cluster IDs Per Route Reflector

The BGP—Multiple Cluster IDs feature allows a route reflector (RR) to belong to multiple clusters, and therefore have multiple cluster IDs. An RR can have a cluster ID configured on a global basis and a per-neighbor basis. A single cluster ID can be assigned to two or more iBGP neighbors. Prior to this feature, an RR had a single, global cluster ID, which was configured by the **bgp cluster-id** router configuration command.

When a cluster ID is configured per neighbor (by the **neighbor cluster-id** router configuration command), the following two changes occur:

- The loop prevention mechanism based on the CLUSTER_LIST attribute is automatically modified to take into account multiple cluster IDs.
- The network administrator can disable client-to-client route reflection based on cluster ID, which allows the network design to change.

The loop prevention mechanism and the CLUSTER_LIST propagation rules are described in the section "How a CLUSTER_LIST Attribute is Used." Disabling client-to-client reflection is described in the section "Behaviors When Disabling Client-to-Client Route Reflection."

How a CLUSTER_LIST Attribute is Used

The CLUSTER_LIST propagation rules differ among releases, depending on whether the device is running a Cisco software release generated before or after the BGP—Multiple Cluster IDs feature was implemented. The same is true for loop prevention based on the CLUSTER_LIST.

The CLUSTER_LIST behavior is described below. Classic refers to the behavior of software released before the multiple cluster IDs feature was implemented; MCID refers to the behavior of software released after the feature was implemented.

CLUSTER_LIST Propagation Rules

- Classic—Before reflecting a route, the RR appends the global cluster ID to the CLUSTER_LIST. If the received route had no CLUSTER_LIST attribute, the RR creates a new CLUSTER_LIST attribute with that global cluster ID.
- MCID—Before reflecting a route, the RR appends the cluster ID of the neighbor the route was received from to the CLUSTER_LIST. If the received route had no CLUSTER_LIST attribute, the RR creates a new CLUSTER_LIST attribute with that cluster ID. This behavior includes a neighbor that is not a client of the speaker. If the nonclient neighbor the route was received from does not have an associated cluster ID, the RR uses the global cluster ID.

Loop Prevention Based on CLUSTER_LIST

- Classic—When receiving a route, the RR discards the route if the RR's global cluster ID is contained in the CLUSTER_LIST of the route.
- MCID—When receiving a route, the RR discards the route if the RR's global cluster ID or any of the cluster IDs assigned to any of the iBGP neighbors is contained in the CLUSTER_LIST of the route.

Behaviors When Disabling Client-to-Client Route Reflection

With the introduction of multiple cluster IDs per iBGP neighbor, it is possible to disable route reflection from client to client on the basis of cluster ID. Disabling route reflection allows you to change the network design. A typical (but not required) scenario after disabling route reflection is that clients are fully meshed, so they have to send more updates, and the RR has client-to-client reflection disabled, so that it has to send fewer updates.

You might want to disable route reflection in a scenario similar to the one in the figure below. An RR has several clients [Provider-Edge (PE) routers] with which it has sessions. The iBGP neighbors that should belong to one cluster were assigned the same cluster ID.

Because the PEs belonging to the same cluster are fully meshed (PE1 and PE2 have a session between them; PE3 and PE4 have a session between them), there is no need to reflect the routes between them. That is, routes from PE1 should be forwarded to PE3 and PE4, but not to PE2.

It is important to know that when the software changes reflection state for a given cluster ID, BGP sends an outbound soft refresh to all clients.

Disabling client-to-client route reflection is done differently and has different results, depending on whether the device is running Cisco software generated before or after the multiple cluster IDs feature was implemented. Classic refers to the behavior of software released before the multiple cluster IDs feature was implemented; MCID refers to the behavior of software released after the multiple cluster IDs feature was implemented.

- Classic—When receiving a route from a client, the RR does not reflect it to any other client. Other scenarios for reflection (client-to-nonclient and nonclient-to-client) are maintained. Disabling of route reflection from client to client is usually done when all the clients are fully meshed (the routes are advertised between the clients via that mesh, so there is no need for reflection). The command to disable client-to-client route reflection is entered in router configuration mode (after the **router bgp** command) and it applies globally to all address families: **no bgp client-to-client reflection**
- MCID—When receiving a route from a client, the RR does not reflect it to another client if both clients belong to a cluster for which client-to-client reflection has been disabled. Therefore, route reflection is disabled only intracluster (within the cluster specified). Other cases for reflection (client-to-nonclient, nonclient-to-client, and intercluster) are maintained. This functionality is usually configured when all the clients for a particular cluster are fully meshed among themselves (but not with clients of other clusters). The command to disable client-to-client route reflection for a particular cluster is entered in router configuration mode and it applies globally to all address families:

  no bgp client-to-client reflection intra-cluster cluster-id {any | *cluster-id1 cluster-id2...***}**

  The **any** keyword is used to disable client-to-client reflection for any cluster.

  The Classic, previously released command for disabling all client-to-client reflection is also still available during this post-MCID release timeframe:

  no bgp client-to-client reflection [all]

  (The optional **all** keyword has no effect in either the positive or negative form of the command, and does not appear in configuration files. It is just to remind the network administrator that both intercluster and intracluster client-to-client reflection are enabled or disabled.)

  In summary, after the introduction of the multiple cluster IDs feature, there are three levels of configuration that can disable client-to-client reflection. The software performs them in the following order, from least specific to most specific:

  1. Least specific: **no bgp client-to-client reflection [all]** Disables intracluster and intercluster client-to-client reflection.
  2. More specific: **no bgp client-to-client reflection intra-cluster cluster-id any** Disables intracluster client-to-client reflection for any cluster-id.
  3. Most specific: **no bgp client-to-client reflection intra-cluster cluster-id** *cluster-id1 cluster-id2 ...* Disables intracluster client-to-client reflection for the specified clusters.

When BGP is advertising updates, the software evaluates each level of configuration in order. Once any level of configuration disables client-to-client reflection, no further evaluation of more specific policies is necessary.

Note the results of the base (positive) and negative (**no**) forms of the three commands listed above:

- A negative configuration (that is, with the **no** keyword) overwrites any less specific configuration.
- A positive configuration (that is, without the **no** keyword) will lose out to (default to) what is configured in a less specific configuration.
- Configurations at any level appear in the configuration file only if they are negative.

All levels can be configured independently and all levels appear in the configuration file independently of the configuration of other levels.

Note that negative configuration makes any more specific configuration unnecessary (because even if the more specific configuration is positive, it is not processed after the negative configuration; if the more specific configuration is negative, it is functionally the same as the earlier negative configuration). The following examples illustrate this behavior.

Example 1

no bgp client-to-client reflection

no bgp client-to-client reflection intra-cluster cluster-id any

Intercluster and intracluster reflection are disabled (based on the first command). The second command disables intracluster reflection, but it is unnecessary because intracluster reflection is already disabled by the first command.

Example 2

no bgp client-to-client reflection intra-cluster cluster-id any

bgp client-to-client reflection intra-cluster cluster-id 1.1.1.1

Cluster ID 1.1.1.1 has intracluster route reflection disabled (even though the second command is positive), because the first command is used to evaluate the update. The first command was negative, and once any level of configuration disables client-to-client reflection, no further evaluation is performed.

Another way to look at this example is that the second command, because it is in a positive form, defaults to the behavior of the first command (which is less specific). Thus, the second command is unnecessary.

Note that the second command would not appear in a configuration file because it is not a negative command.

How to Use BGP-Multiple Cluster IDs

Configuring a Cluster ID per Neighbor

Perform this task on an iBGP peer (usually a route reflector) to configure a cluster ID per neighbor. Configuring a cluster ID per neighbor causes the loop-prevention mechanism based on the CLUSTER_LIST to be automatically modified to take into account multiple cluster IDs. Also, you gain the ability to disable client-to-client route reflection on the basis of cluster ID. The software tags the neighbor so that you can disable route reflection with the use of another command. (See the tasks for disabling client-to-client reflection later in this module.)

Note When you change a cluster ID for a neighbor, BGP automatically does an inbound soft refresh and an outbound soft refresh for all iBGP peers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ip-address* | *ipv6-address*} **remote-as** *autonomous-system-number*
5. **neighbor** {*ip-address* | *ipv6-address*} **cluster-id** *cluster-id*
6. **end**
7. **show ip bgp cluster-ids**

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. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**neighbor** {*ip-address* | *ipv6-address*} **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.1.2 remote-as 65000`	Adds an entry to the BGP routing table.
Step 5	**neighbor** {*ip-address* | *ipv6-address*} **cluster-id** *cluster-id*	Assigns a cluster ID to the specified neighbor.

	Command or Action	Purpose
	Example: `Device(config-router)# neighbor 192.168.1.2 cluster-id 0.0.0.1`	• The cluster ID can be in dotted decimal format (such as 192.168.7.4) or decimal format (such as 23), with a maximum of 4 bytes. • A cluster ID that is configured in decimal format (such as 23) is modified to dotted decimal format (such as 0.0.0.23) when it appears in a configuration file. • When you change a cluster ID for a neighbor, BGP automatically does an inbound soft refresh and an outbound soft refresh for all iBGP peers.
Step 6	**end** **Example:** `Device(config-router)# end`	(Optional) Exits to privileged EXEC mode.
Step 7	**show ip bgp cluster-ids** **Example:** `Device# show ip bgp cluster-ids`	(Optional) Lists: • the global cluster ID (whether configured or not) • all cluster IDs that are configured to a neighbor • all cluster IDs for which the network administrator has disabled reflection

Disabling Intracluster and Intercluster Client-to-Client Reflection

Perform the following task on a route reflector if you want to disable both intracluster and intercluster client-to-client reflection. Doing so is the broadest (least specific) way to disable client-to-client reflection. Before advertising updates, the software evaluates each level of configuration in order from least specific to most specific. Once any level of configuration disables client-to-client reflection, no further evaluation of more specific policies is needed.

Note When the software changes reflection state for a given cluster ID, BGP sends an outbound soft refresh to all clients.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp client-to-client reflection** [**all**]

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. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**no bgp client-to-client reflection** [**all**] **Example:** `Device(config-router)# no bgp client-to-client reflection all`	Disables intracluster and intercluster client-to-client route reflection. • The **all** keyword is just to emphasize that the **bgp client-to-client reflection** command affects both intracluster and intercluster reflection; the **all** keyword has no effect in the positive or negative form of the command.

Disabling Intracluster Client-to-Client Reflection for Any Cluster ID

Perform the following task on a route reflector to disable intracluster client-to-client reflection for any cluster ID. Doing so is considered to be the middle of the three levels of commands available to disable client-to-client reflection. That is, it is more specific than disabling intracluster and intercluster client-to-client reflection, but it is not as specific as disabling intracluster client-to-client reflection for certain cluster IDs.

Before advertising updates, the software evaluates each level of configuration in order from least specific to most specific. Once any level of configuration disables client-to-client reflection, no further evaluation of more specific policies is needed.

Note When the software changes reflection state for a given cluster ID, BGP sends an outbound soft refresh to all clients.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp client-to-client reflection intra-cluster cluster-id any**

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. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**no bgp client-to-client reflection intra-cluster cluster-id any** **Example:** `Device(config-router)# no bgp client-to-client reflection intra-cluster cluster-id any`	Disables intracluster client-to-client route reflection for any cluster.

Disabling Intracluster Client-to-Client Reflection for Specified Cluster IDs

Perform the following task on a route reflector to disable intracluster client-to-client reflection for specified cluster IDs. Doing so is considered to be the most specific of the three levels of commands available to disable client-to-client reflection. Before advertising updates, the software evaluates each level of configuration in order from least specific to most specific. Once any level of configuration disables client-to-client reflection, no further evaluation of more specific policies is needed.

Note When the software changes reflection state for a given cluster ID, BGP sends an outbound soft refresh to all clients.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **no bgp client-to-client reflection intra-cluster cluster-id** *cluster-id1* [*cluster-id2...*]

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. • The *as-number* argument indicates the number of an autonomous system that identifies the router to other BGP routers and tags the routing information passed along. Valid numbers are from 0 to 65535. Private autonomous system numbers that can be used in internal networks range from 64512 to 65535.
Step 4	**no bgp client-to-client reflection intra-cluster cluster-id** *cluster-id1* [*cluster-id2...*] **Example:** `Device(config-router)# no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.1 0.0.0.3 105`	Disables intracluster client-to-client route reflection within each of the specified clusters. • Note that this example command will appear in the configuration file as "no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.1 0.0.0.3 0.0.0.105" because decimal cluster ID numbers appear in the dotted decimal format.

Configuration Examples for BGP-Multiple Cluster IDs

Example: Per-Neighbor Cluster ID

The following example is configured on a route reflector. The neighbor (client) at IPv6 address 2001:DB8:1::1 is configured to have the cluster ID of 0.0.0.6:

```
router bgp 6500
```

```
 neighbor 2001:DB8:1::1 cluster-id 0.0.0.6
```

Example: Disabling Client-to-Client Reflection

The following example disables all intracluster and intercluster client-to-client reflection:

```
router bgp 65000
 no bgp client-to-client reflection all
```

The following example disables intracluster client-to-client reflection for any cluster ID:

```
router bgp 65000
 no bgp client-to-client reflection intra-cluster cluster-id any
```

The following example disables intracluster client-to-client reflection for the specified cluster IDs 0.0.0.1, 14, 15, and 0.0.0.6:

```
router bgp 65000
 no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.1 14 15 0.0.0.6
```

Remember that a cluster ID specified in the **neighbor cluster-id** command in decimal format (such as 23) will appear in a configuration file in dotted decimal format (such as 0.0.0.23). The decimal format does not appear in the configuration file. The running configuration might look like this:

```
router bgp 65000
no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.1
no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.6
no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.14
no bgp client-to-client reflection intra-cluster cluster-id 0.0.0.15
```

Additional References

Related Documents

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

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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-Multiple Cluster IDs

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 68: Feature Information for BGP—Multiple Cluster IDs

Feature Name	Releases	Feature Information
BGP—Multiple Cluster IDs		The BGP—Multiple Cluster IDs feature allows an iBGP neighbor (usually a route reflector) to have multiple cluster IDs: a global cluster ID and additional cluster IDs that are assigned to clients (neighbors). Prior to the introduction of this feature, a device could have a single, global cluster ID. When a network administrator configures per-neighbor cluster IDs: • The loop prevention mechanism based on a CLUSTER_LIST is automatically modified to take into account multiple cluster IDs. • A network administrator can disable client-to-client route reflection based on cluster ID. The following commands were introduced: • **bgp client-to-client reflection intra-cluster** • **neighbor cluster-id** • **show ip bgp cluster-ids** The following commands were modified: • **bgp client-to-client reflection** • **show ip bgp neighbors** • **show ip bgp template peer-session** • **show ip bgp update-group**

CHAPTER 56

Multicast VPN BGP Dampening

A single receiver in a specific multicast group or a group of receivers that are going up and down frequently and interested in a specific multicast group activates the Multicast VPN BGP Dampening feature to dampen type 7 routes (C-multicast route Join/Prune) within the core using BGP signaling. The feature reduces the churn caused by customer-side join/prune requests to avoid unnecessary BGP MVPN type 6/7 C-route control information.

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 Multicast VPN BGP Dampening

- You understand the concepts in the "BGP Route Dampening" module of the *IP Routing: BGP Configuration Guide*.

Information About Multicast VPN BGP Dampening

Overview of Multicast VPN BGP Dampening

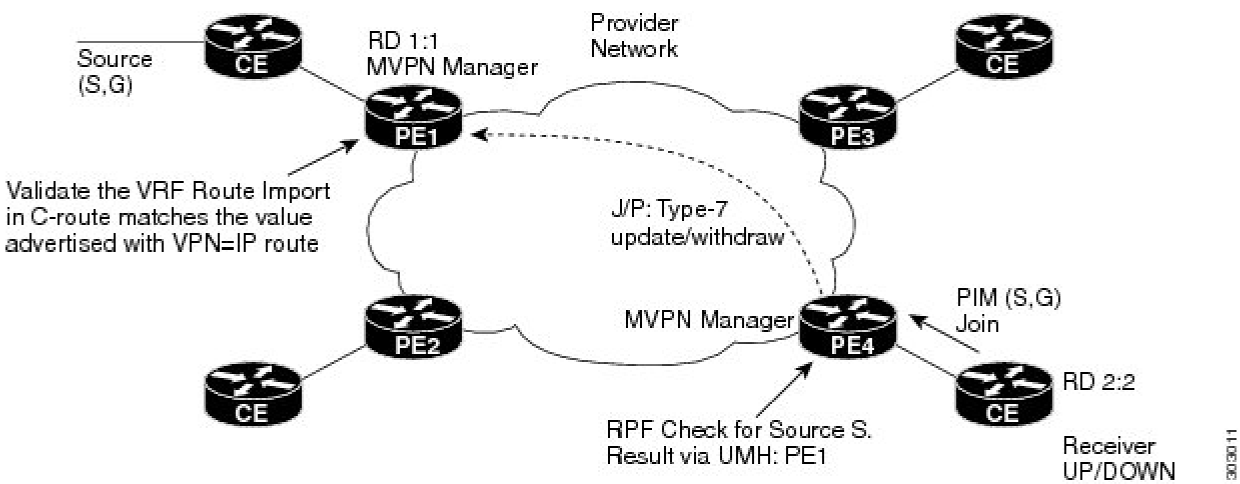
BGP Route Dampening

Route dampening is a BGP feature designed to minimize the propagation of flapping routes across an internetwork. A route is considered to be flapping when its availability alternates repeatedly. Cisco devices that are running BGP contain a mechanism designed to “dampen” the destabilizing effect of flapping routes. When a Cisco device running BGP detects a flapping route, it automatically dampens that route.

The figure below shows illustrates the Multicast VPN BGP dampening mechanism.

Multicast VPN BGP Dampening

Figure 74: Multicast VPN BGP Dampening



A single receiver in a multicast group or a group of receivers that are flapping frequently and interested in a specific multicast group actrivates multicast VPN (MVPN) BGP dampening. MVPN BGP dampening dampens the type 7 multicast routes (customer-multicast, or “C-multicast,” route join/prune) within the core using BGP signaling.

When MVPN BGP dampening is not enabled, the source sends data even though the receiver may be down. When the receiver is down, there is no periodic 60-second C-PIM join towards the provider edge (PE) device causing the PIM to timeout on the PE side after the default period (three minutes). The MVPN manager sends a prune message to BGP, which is a type 7 route (C-multicast route withdraw).

When the receiver is up, it sends a new (S,G) join request to the customer edge (CE) device. The C-PIM join is received by the PE device and a new type 7 C-multicast update is sent by BGP to the auto-discovered MVPN peers. The upstream multicast peer converts the BGP type 7 update to a PIM join to the source, and the source sends the data traffic that the receiver should receive via the downstream PE using the MDT tunnel. If the receiver goes up and down frequently, the source side PIM receives join/prune messages frequently and can cause the source to respond accordingly.

When MVPN BGP dampening is enabled, the general dampening mechanism in BGP will be applied to MVPN VRF instances. Join/Prune messages from the CE side are sent from an MVPN manager as updates/withdraw to the MVPN PE device. The MVPN manager on PE devices send join/prune messages to the customer side for Reverse Path Forwarding (RPF) and upstream multihop (UMH) nexthop changes.

How to Configure Multicast VPN BGP Dampening

Configuring Multicast VPN BGP Dampening

Perform this task to enable and configure multicast VPN BGP dampening.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family** [**ipv4** | **ipv6**] **mvpn vrf** *vrf-name*
5. **bgp dampening** [*half-life reuse suppress max-suppress-time*]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 45000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family** [**ipv4** | **ipv6**] **mvpn vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 mvpn vrf blue`	Specifies the address family and enters address family configuration mode. • Use the **ipv4** keyword to enable IPv4 multicast C-route exchange. • Use the **ipv6** keyword to enable IPv6 multicast C-route exchange.

	Command or Action	Purpose
		Note The **vrf** keyword and *vrf-name* argument must be specified at this point to enable multicast VPN BGP dampening in the next step.
Step 5	**bgp dampening** [*half-life reuse suppress max-suppress-time*] **Example:** `Device(config-router-af)# bgp dampening 30 1500 10000 120`	Enables BGP route dampening and changes the default values of route dampening factors. The *half-life*, *reuse*, *suppress*, and *max-suppress-time* arguments are all position dependent; if one argument is entered, then all the arguments must be entered. **Note** Repeat steps 4 and 5 to enable multicast VPN BGP dampening on alternative VRFs.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and enters privileged EXEC mode.

Monitoring and Maintaining Multicast VPN BGP Dampening

Perform the steps in this task as required to monitor and maintain multicast VPN BGP dampening.

SUMMARY STEPS

1. **enable**
2. **show bgp** {**ipv4** | **ipv6**} **mvpn** {**all** | **rd** *route-distinguisher* | **vpn** *vrf-name*} [**dampening** {**dampened-paths** | **flap-statistics** [**filter-list** *access-list-number* | **quote-regexp** *regexp* | **regexp** *regexp*]}]
3. **clear ip bgp** {**ipv4** | **ipv6**} **mvpn vrf** *vrf-name* {**dampening** | **flap-statistics**}

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 **show bgp** {**ipv4** | **ipv6**} **mvpn** {**all** | **rd** *route-distinguisher* | **vpn** *vrf-name*} [**dampening** {**dampened-paths** | **flap-statistics** [**filter-list** *access-list-number* | **quote-regexp** *regexp* | **regexp** *regexp*]}]

Use this command to monitor multicast VPN BGP dampening.

- The **dampened-path** keyword displays information about BGP dampened routes.
- The **parameters** keyword displays detailed BGP dampening information.
- The **flap-statistics** keyword displays information on BGP flap statistics.

Example:

```
Device# show bgp ipv4 mvpn vrf blue route-type 7 111.111.111.111:11111 55 202.100.0.6 232.1.1.1

BGP routing table entry for [7][111.111.111.111:11111][55][202.100.0.6/32][232.1.1.1/32]/22, version
 17
Paths: (1 available, no best path)
Flag: 0x820
  Not advertised to any peer
  Refresh Epoch 1
  Local, (suppressed due to dampening)
    0.0.0.0 from 0.0.0.0 (205.3.0.3)
      Origin incomplete, localpref 100, weight 32768, valid, sourced, local
      Extended Community: RT:205.1.0.1:1
      Dampinfo: penalty 3472, flapped 4 times in 00:04:42, reuse in 00:00:23
      rx pathid: 0, tx pathid: 0
```

Step 3 **clear ip bgp** {**ipv4** | **ipv6**} **mvpn vrf** *vrf-name* {**dampening** | **flap-statistics**}

Use this command to clear the accumulated penalty for routes that are received on a router that has multicast VPN BGP dampening enabled.

- The **dampening** keyword clears multicast VPN BGP dampening information.
- The **flap-statistic** keyword clears multicast VPN BGP dampening flap statistics.

Example:

```
Device# clear ip bgp ipv4 mvpn vrf blue dampening
```

Configuration Examples for Multicast VPN BGP Dampening

Example: Configuring Multicast VPN BGP Dampening

The following example shows multicast VPN BGP dampening is applied to the VRFs named blue and red, but not to the VRF named green:

```
address-family ipv4 mvpn vrf blue
 bgp dampening

address-family ipv4 mvpn vrf red
 bgp dampening

address-family ipv4 mvpn vrf green
 no bgp dampening
```

Additional References for Multicast VPN BGP Dampening

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
BGP route dampening	"BGP Route Dampening" section of the "Configuring Internal BGP Features" module in the *IP Routing: BGP Configuration Guide*

Standards and RFCs

Standard/RFC	Title
RFC 2439	*BGP Route Flap Damping*

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 Multicast VPN BGP Dampening

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 69: Feature Information for Multicast VPN BGP Dampening

Feature Name	Releases	Feature Information
Multicast VPN BGP Dampening	Cisco IOS XE Release 3.8S	A single receiver in a specific multicast group or a group of receivers that are going up and down frequently and interested in a specific multicast group will cause the Multicast VPN BGP Dampening feature to dampen type 7 routes (C-multicast route join/prune) within the core using BGP signaling. The following commands were introduced or modified: **address-family mvpn**, **clear ip bgp mvpn**, **show bgp mvpn**, and **show ip bgp ipv4**.

CHAPTER 57

BGP-VPN Distinguisher Attribute

The BGP—VPN Distinguisher Attribute feature allows a network administrator to keep source route targets (RTs) private from an Autonomous System Border Router (ASBR) in a destination autonomous system. An RT at an egress ASBR is mapped to a VPN distinguisher, the VPN distinguisher is carried through the eBGP, and then it is mapped to an RT at the ingress ASBR.

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.

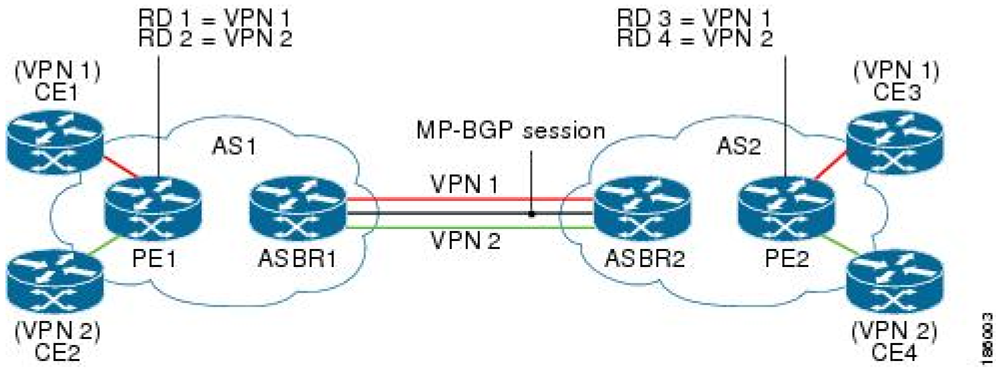
Information About BGP-VPN Distinguisher Attribute

Role and Benefit of the VPN Distinguisher Attribute

Route-target (RT) extended community attributes identify the VPN membership of routes. The RT attributes are placed onto a route at the exporting (egress) provider edge router (PE) and are transported across the iBGP cloud and across autonomous systems. Any Virtual Routing and Forwarding (VRF) instances at the remote PE that want to import such routes must have the corresponding RTs set as import RTs for that VRF.

The figure below illustrates two autonomous systems, each containing customer edge routers (CEs) that belong to different VPNs. Each PE tracks which route distinguisher (RD) corresponds to which VPN, thus controlling the traffic that belongs to each VPN.

Figure 75: Scenario in Which ASBRs Translate RTs Between Autonomous Systems



In an Inter-AS Option B scenario like the one in the figure above, these routes are carried across an AS boundary from Autonomous System Border Router 1 (ASBR1) to ASBR2 over an MP-eBGP session, with the routes' respective RTs as extended community attributes being received by ASBR2.

ASBR2 must maintain complex RT mapping schemes to translate RTs originated by AS1 to RTs recognized by AS2, so that the RTs can be imported by their respective VPN membership CE connections on PE2 for CE3 and CE4.

Some network administrators prefer to hide the RTs they source in AS1 from devices in AS2. In order to do that, the administrator must differentiate routes belonging to each VPN with a certain attribute so that the RTs can be removed on the outbound side of ASBR1 before sending routes to ASBR2, and ASBR2 can then map that attribute to recognizable RTs in AS2. The VPN Distinguisher (VD) extended community attribute serves that purpose.

The benefit of the BGP—VPN Distinguisher Attribute feature is that source RTs can be kept private from devices in destination autonomous systems.

How the VPN Distinguisher Attribute Works

The network administrator configures the egress ASBR to perform translation of RTs to a VPN distinguisher extended community attribute, and configures the ingress ASBR to perform translation of the VPN distinguisher to RTs. More specifically, the translation is achieved as follows:

On the Egress ASBR

- An outbound route map specifies a **match excommunity** clause that determines which VPN routes are subject to mapping, based on the route's RT values.
- A **set extcommunity vpn-distinguisher** command sets the VPN distinguisher that replaces the RTs.
- The **set extcomm-list delete** command that references the same set of RTs is configured to remove the RTs, and then the route is sent to the neighboring ingress ASBR.

On the Ingress ARBR

- An inbound route map specifies a **match excommunity vpn-distinguisher** command that determines which VPN routes are subject to mapping, based on the route's VPN distinguisher.
- The **set extcommunity rt** command specifies the RTs that replace the VPN distinguisher.
- For routes that match the clause, the VPN distinguisher is replaced with the configured RTs.

Additional Behaviors Related to the VPN Distinguisher

On the egress ASBR, if a VPN route matches a route map clause that does not have the **set extcommunity vpn-distinguisher** command configured, the RTs that the VPN route is tagged with are retained.

The VPN distinguisher is transitive across the AS boundary, but is not carried within the iBGP cloud. That is, the ingress ASBR can receive the VPN distinguisher from an eBGP peer, but the VPN distinguisher is discarded on the inbound side after it is mapped to the corresponding RTs.

On the ingress ASBR, if a VPN route carrying the VPN distinguisher matches a route map clause that does not have a **set extcommunity rt** command configured in the inbound route map, the system does not discard the attribute, nor does it propagate the attribute within the iBGP cloud. The VPN distinguisher for the route is retained so that the network administrator can configure the correct inbound policy to translate the VPN distinguisher to the RTs that the VPN route should carry. If the route is sent to eBGP peers, the VPN distinguisher is carried as is. The network administrator could configure a route-map entry to remove the VPN distinguisher from routes sent to eBGP peers.

Configuring a **set extcommunity vpn-distinguisher** command in an outbound route map or a **match excommunity** command in an inbound route map results in an outbound or inbound route refresh request, respectively, in order to update the routes being sent or received.

How to Configure BGP-VPN Distinguisher Attribute

Replacing an RT with a VPN Distinguisher Attribute

Perform this task on an egress ASBR to replace a route target (RT) with a VPN distinguisher extended community attribute. Remember to replace the VPN distinguisher with a route target on the ingress ASBR; that task is described in the “Replacing a VPN Distinguisher Attribute with an RT” section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip extcommunity-list** *expanded-list* {**permit** | **deny**} **rt** *value*
4. **exit**
5. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
6. **match extcommunity** *extended-community-list-name*
7. **set extcomm-list** *extcommunity-name* **delete**
8. **set extcommunity vpn-distinguisher** *id*
9. **exit**
10. **route-map** *map-name* {**permit** | **deny**} [*sequence-number*]
11. **exit**
12. **router bgp** *as-number*
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **address-family vpnv4**
15. **neighbor** *ip-address* **activate**
16. **neighbor** *ip-address* **route-map** *map-name* **out**
17. **exit-address-family**

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* {**permit** | **deny**} **rt** *value* **Example:** `Device(config)# ip extcommunity-list 4 permit rt 101:100`	Configures an IP extended community list to configure Virtual Private Network (VPN) route filtering, such that routes with the specified RT are in the extended community list. • This example permits routes having RT 101:100 into the extended community list 4.
Step 4	**exit** **Example:** `Device(config-extcomm-list)# exit`	Exits the configuration mode and enters the next higher configuration mode.
Step 5	**route-map** *map-tag* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map vpn-id-map1 permit 10`	Configures a route map that permits or denies the routes allowed by the subsequent **match** command. • This example permits the routes allowed by the subsequent **match** command.
Step 6	**match extcommunity** *extended-community-list-name* **Example:** `Device(config-route-map)# match extcommunity 4`	Matches on the specified community list. • For this example, routes that match the extended community list 4 (which was configured in Step 3) are subject to the subsequent **set** commands.
Step 7	**set extcomm-list** *extcommunity-name* **delete** **Example:** `Device(config-route-map)# set extcomm-list 4 delete`	Deletes the RT from routes that are in the specified extended community list. • For this example, RTs are deleted from routes that are in extended community list 4.
Step 8	**set extcommunity vpn-distinguisher** *id* **Example:** `Device(config-route-map)# set extcommunity vpn-distinguisher 111:100`	For the routes that are permitted by the route map, sets the specified VPN distinguisher. • For this example, routes that match extended community 4 have their VPN distinguisher set to 111:100.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 10	**route-map** *map-name* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map vpn-id-map1 permit 20`	(Optional) Configures a route map entry that permits routes. • This example configures a route map entry that permits other routes not subject to the RT-to-VPN distinguisher mapping. If you do not perform this step, all other routes are subject to an implicit deny.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 12	**router bgp** *as-number* **Example:** `Device(config)# router bgp 2000`	Enters router configuration mode and creates a BGP routing process.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.101.1 remote-as 2000`	Specifies that the neighbor belongs to the autonomous system.
Step 14	**address-family vpnv4** **Example:** `Device(config-router)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address family-specific configurations.
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.101.1 activate`	Activates the specified neighbor.
Step 16	**neighbor** *ip-address* **route-map** *map-name* **out** **Example:** `Device(config-router-af)# neighbor 192.168.101.1 route-map vpn-id-map1 out`	Applies the specified outgoing route map to the specified neighbor.

	Command or Action	Purpose
Step 17	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters privileged EXEC mode.

Replacing a VPN Distingusher Attribute with an RT

Perform this task on an ingress ASBR to replace a VPN distinguisher extended community attribute with a route target (RT) attribute. This task assumes you already configured the egress ASBR to replace the RT with a VPN distinguisher; that task is described in the "Replacing an RT with a VPN Distinguisher Attribute" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip extcommunity-list** *expanded-list* {**permit** | **deny**} **vpn-distinguisher** *id*
4. **exit**
5. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
6. **match extcommunity** *extended-community-list-name*
7. **set extcomm-list** *extcommunity-name* **delete**
8. **set extcommunity rt** *value* **additive**
9. **exit**
10. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
11. **exit**
12. **router bgp** *as-number*
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **address-family vpnv4**
15. **neighbor** *ip-address* **activate**
16. **neighbor** *ip-address* **route-map** *map-name* **in**
17. **exit-address-family**

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	**ip extcommunity-list** *expanded-list* {**permit** | **deny**} **vpn-distinguisher** *id* **Example:** `Device(config)# ip extcommunity-list 51 permit vpn-distinguisher 111:100`	Configures an IP extended community list to configure Virtual Private Network (VPN) route filtering, such that routes with the specified VPN distinguisher are in the extended community list. • This example permits routes having VPN distinguisher 111:110 into the extended community list 51.
Step 4	**exit** **Example:** `Device(config-extcomm-list)# exit`	Exits the configuration mode and enters the next higher configuration mode.
Step 5	**route-map** *map-tag* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map vpn-id-rewrite-map1 permit 10`	Configures a route map that permits or denies the routes allowed by the subsequent **match** command. • This example permits the routes allowed by the subsequent **match** command.
Step 6	**match extcommunity** *extended-community-list-name* **Example:** `Device(config-route-map)# match extcommunity 51`	Matches on the specified community list. • For this example, routes that match the extended community list 51 (which was configured in Step 3) are subject to the subsequent **set** commands.
Step 7	**set extcomm-list** *extcommunity-name* **delete** **Example:** `Device(config-route-map)# set extcomm-list 51 delete`	Deletes the VPN distinguisher from routes that are in the specified extended community list. • For this example, VPN distinguishers are deleted from routes that are in extended community list 51.
Step 8	**set extcommunity rt** *value* **additive** **Example:** `Device(config-route-map)# set extcommunity rt 101:1 additive`	Sets the routes that are permitted by the route map with the specified RT. • For this example, routes that match extended community 51 have their RT set to 101:1. The **additive** keyword causes the RT to be added to the RT list without replacing any RTs.
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:**	(Optional) Configures a route map entry that permits routes. • This example configures a route map entry that permits other routes not subject to the VPN

	Command or Action	Purpose
	`Device(config)# route-map vpn-id-rewrite-map1 permit 20`	distinguisher-to-RT mapping. If you do not perform this step, all other routes are subject to an implicit deny.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 12	**router bgp** *as-number* **Example:** `Device(config)# router bgp 3000`	Enters router configuration mode and creates a BGP routing process.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.0.81 remote-as 3000`	Specifies that the neighbor belongs to the autonomous system.
Step 14	**address-family vpnv4** **Example:** `Device(config-router-af)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address family-specific configurations.
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.0.81 activate`	Activates the specified neighbor.
Step 16	**neighbor** *ip-address* **route-map** *map-name* **in** **Example:** `Device(config-router-af)# neighbor 192.168.0.81 route-map vpn-id-rewrite-map1 in`	Applies the specified outgoing route map to the specified neighbor.
Step 17	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters privileged EXEC mode.

Example

Configuration Examples for BGP-VPN Distinguisher Attribute

Example: Translating RT to VPN Distinguisher to RT

The following example shows the egress ASBR configuration to replace a route target (RT) with a VPN distinguisher, and shows the ingress ASBR configuration to replace the VPN distinguisher with a route target.

On the egress ASBR, IP extended community list 1 is configured to filter VPN routes by permitting only routes with RT 101:100. A route map named vpn-id-map1 says that any route that matches on routes that are allowed by IP extended community list 1 are subject to two **set** commands. The first **set** command deletes the RT from the route. The second **set** command sets the VPN distinguisher attribute to 111:100.

The **route-map vpn-id-map1 permit 20** command allows other routes, which are not part of the RT-to-VPN distinguisher mapping, to pass the route map so that they are not discarded. Without this command, the implicit deny would cause these routes to be discarded.

Finally, in autonomous system 2000, for the VPNv4 address family, the route map vpn-id-map1 is applied to routes going out to the neighbor at 192.168.101.1.

Egress ASBR

```
ip extcommunity-list 1 permit rt 101:100
!
route-map vpn-id-map1 permit 10
 match extcommunity 1
 set extcomm-list 1 delete
 set extcommunity vpn-distinguisher 111:100
!
route-map vpn-id-map1 permit 20
!
router bgp 2000
 neighbor 192.168.101.1 remote-as 2000
 address-family vpnv4
  neighbor 192.168.101.1 activate
  neighbor 192.168.101.1 route-map vpn-id-map1 out
  exit-address-family
!
```

On the ingress ASBR, IP extended community list 51 allows routes with a VPN distinguisher of 111:100. A route map named vpn-id-rewrite-map1 says that any route that matches on routes that are allowed by IP extended community list 51 are subject to two **set** commands. The first **set** command deletes the VPN distinguisher from the route. The second **set** command sets the RT to 101:1, and that RT is added to the RT list without replacing any RTs.

The **route-map vpn-id-rewrite-map1 permit 20** command allows other routes, which are not part of the VPN distinguisher-to-RT mapping, to pass the route map so that they are not discarded. Without this command, the implicit deny would cause those routes to be discarded.

Finally, in autonomous system 3000, for the VPNv4 address family, the route map named vpn-id-rewrite-map1 is applied to incoming routes destined for the neighbor at 192.168.0.81.

Ingress ASBR

```
ip extcommunity-list 51 permit vpn-distinguisher 111:100
!
route-map vpn-id-rewrite-map1 permit 10
 match extcommunity 51
 set extcomm-list 51 delete
 set extcommunity rt 101:1 additive
!
route-map vpn-id-rewrite-map1 permit 20
!
router bgp 3000
 neighbor 192.168.0.81 remote-as 3000
 address-family vpnv4
  neighbor 192.168.0.81 activate
  neighbor 192.168.0.81 route-map vpn-id-rewrite-map1 in
  exit-address-family
!
```

Additional References

Related Documents

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

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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-VPN Distinguisher Attribute

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 70: Feature Information for BGP—VPN Distinguisher Attribute

Feature Name	Releases	Feature Information
BGP—VPN Distinguisher Attribute		The BGP—VPN Distinguisher Attribute feature allows a network administrator to keep source RTs private from an ASBR in a destination autonomous system. An RT at an egress ASBR is mapped to a VPN distinguisher, the VPN distinguisher is carried through the eBGP, and then it is mapped to an RT at the ingress ASBR. The following command was introduced: • **set extcommunity vpn-distinguisher** The following command was modified: • **show ip bgp vpnv4**

CHAPTER 58

BGP—Support for iBGP Local-AS

Prior to the BGP—Support for iBGP Local-AS feature, the **neighbor local-as** command was used on a BGP speaker to change the AS negotiated for an eBGP neighbor and to modify the AS_PATH sent and/or received. The **neighbor local-as** command can now be used to do the same on an iBGP session. AS negotiation creates an iBGP session and we enable sending iBGP attributes (LOCAL_PREF, ORIGINATOR_ID, and CLUSTER_LIST) over it, and accept this attributes when received from this session. This functionality is useful when merging two autonomous systems into one.

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 Support for iBGP Local-AS

- This feature is not supported for a peer that belongs to a confederation.
- Nonlocal-AS iBGP neighbors that are in a single AS are put into a separate update group from iBGP neighbors that are configured with the iBGP Local-AS feature.

- Two iBGP neighbors that are in two different autonomous systems and that are configured as iBGP Local-AS neighbors are put into separate update groups.

Information About Support for iBGP Local-AS

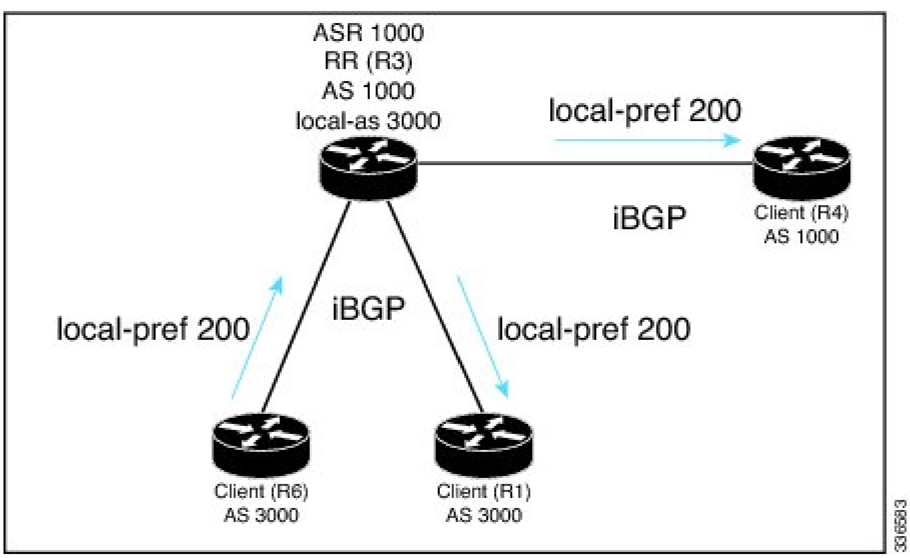
Support for iBGP Local-AS

Prior to the Support for iBGP Local-AS feature, when a peer (or peer group) was configured with the **neighbor local-as** command and the **neighbor remote-as** command that specified the same AS number, the session would be negotiated as an iBGP session (this happens when the advertised ASes in both OPEN messages are the same). However, updates were propagated as in an eBGP session (LOCAL_PREF, ORIGINATOR_ID and CLUSTER_LIST were not propagated), and could cause errors if they were received via this session. Thus, iBGP local-AS was not fully supported.

The Support for iBGP Local-AS feature means all those iBGP attributes are propogated. Additionally, as in any iBGP session, the AS is not prepended in AS_PATH attribute when advertising routes to an iBGP local-as session.

The figure below illustrates a scenario where this feature is being used to facilitate the merging of two autonomous systems. The route reflector R3 and R4 belong to AS 1000; R1 and R6 belong to AS 3000. The RR is configured with **neighbor local-as 3000** and **neighbor remote-as 3000** commands. Even though the routers belong to two different autonomous systems, attributes like the LOCAL_PREF are preserved in the updates from R6 to R4 and R6 to R1 (as show in the figure), and also in the updates from R4 to R1 and R4 to R6 (not shown in the figure).

Figure 76: Support for iBGP Local-AS to Preserve iBGP Policies Between Two Autonomous Systems



Benefits of iBGP Local-AS

This feature is used when merging two ISPs that have different autonomous system numbers. It is desirable to preserve attributes that are considered internal (LOCAL_PREF, ORIGINATOR_ID, and CLUSTER_LIST) in the routes that are being propogated to other autonomous system.

How to Configure iBGP Local-AS

Configuring iBGP Local-AS

Configure the iBGP Local-AS feature on a BGP speaker for a given neighbor when you want that session to behave as a full iBGP session. This configuration is typically performed on a route reflector, but not exclusively on it. In a route reflector you can optionally configure changing iBGP attributes sent to a neighbor via the command **allow-policy** (this command is not exclusive for this feature and can be used on any RR).

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 unicast-routing**
4. **router bgp** *autonomous-system-number*
5. **neighbor** *peer-group-name* **peer-group**
6. **neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name*
7. **neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **remote-as** *as-number*
8. **neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **local-as** *as-number*
9. **neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **route-reflector-client**
10. **address-family vpnv4**
11. **neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **allow-policy**
12. **exit**
13. **address-family vpnv6**
14. **neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **allow-policy**
15. **end**
16. **show ip bgp vpnv4 all neighbors** {*ip-address* | *ipv6-address*} **policy**
17. **show ip bgp vpnv4 all update-group** *update-group*
18. **show ip bgp vpnv4 all neighbors** {*ip-address* | *ipv6-address*}

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	**ipv6 unicast-routing** **Example:** `Device(config)# ipv6 unicast-routing`	Enables the forwarding of IPv6 unicast datagrams.
Step 4	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 1000`	Enters router configuration mode to create or configure a BGP routing process.
Step 5	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-router)# neighbor rr-client-ab peer-group`	(Optional) Identifies a peer group.
Step 6	**neighbor** {*ip-address* | *ipv6-address*} **peer-group** *peer-group-name* **Example:** `Device(config-router)# neighbor 192.168.3.3 peer-group rr-client-ab`	(Optional) Configures a BGP neighbor to be a member of a peer group.
Step 7	**neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **remote-as** *as-number* **Example:** `Device(config-router)# neighbor rr-client-ab remote-as 3000`	Identifies the AS of the neighbor or peer group.
Step 8	**neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **local-as** *as-number* **Example:** `Device(config-router)# neighbor rr-client-ab local-as 3000`	Configures the local-AS feature for the neighbor or peer group.
Step 9	**neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **route-reflector-client** **Example:** `Device(config-router)# neighbor rr-client-ab route-reflector-client`	Configures the local device to be a route reflector and configures the neighbor or peer group to be its client.
Step 10	**address-family vpnv4** **Example:** `Device(config-router)# address-family vpnv4`	(Optional) Places the router in VPNv4 address family configuration mode.

	Command or Action	Purpose
Step 11	**neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **allow-policy** **Example:** `Device(config-router-af)# neighbor rr-client-ab allow-policy`	(Optional) Allows the RR to be configured to change iBGP attributes for the specified neighbor or peer group.
Step 12	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and enters router configuration mode.
Step 13	**address-family vpnv6** **Example:** `Device(config-router)# address-family vpnv6`	(Optional) Places the router in VPNv6 address family configuration mode.
Step 14	**neighbor** {*ip-address* | *ipv6-address* | *peer-group*} **allow-policy** **Example:** `Device(config-router-af)# neighbor rr-client-ab allow-policy`	(Optional) Allows the RR to be configured to change iBGP attributes for the specified neighbor or peer group.
Step 15	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode, and enters privileged EXEC mode.
Step 16	**show ip bgp vpnv4 all neighbors** {*ip-address* | *ipv6-address*} **policy** **Example:** `Device# show ip bgp vpnv4 all neighbors 192.168.3.3 policy`	(Optional) Displays the locally configured policies of the neighbor. • The output includes the phrase “allow-policy” if the **neighbor allow-policy** command was configured for that neighbor.
Step 17	**show ip bgp vpnv4 all update-group** *update-group* **Example:** `Device# show ip bgp vpnv4 all update-group 2`	(Optional) Displays the information for the update group. • The output includes the phrase “Allow-policy” if the **neighbor allow-policy** command was configured for neighbors in the update group.
Step 18	**show ip bgp vpnv4 all neighbors** {*ip-address* | *ipv6-address*} **Example:** `Device# show ip bgp vpnv4 all neighbors 192.168.3.3`	(Optional) Displays information about the neighbor. • The output includes the remote AS and local AS, which will indicate the same AS number when the Support for iBGP Local-AS feature is configured.

Configuration Examples for iBGP Local-AS

Example: Configuring iBGP Local-AS

The example configures a route reflector (RR) in AS 4000 to treat BGP sessions with the peer group rr-client-2 in AS 2500 as iBGP sessions. That is, iBGP attributes (LOCAL_PREF, ORIGINATOR_ID, and CLUSTER_LIST) will not be dropped from routes in advertisements to and from the neighbors belonging to the peer group; the attributes will be passed unmodified. AS 2500 will not be prepended to the AS_PATH attribute in routes to or from the peer group.

Additionally, the **neighbor allow-policy** command configures that the network administrator can configure iBGP policies on the RR. That is, an outbound route map can be configured to change attributes that are sent to the downstream peers. In this example, the command is applied to VPNv4 and VPNv6 address families.

```
router bgp 4000
 neighbor rr-client-2 peer-group
 neighbor 192.168.1.1 peer-group rr-client-2
 neighbor 192.168.4.1 peer-group rr-client-2
 neighbor rr-client-2 remote-as 2500
 neighbor rr-client-2 local-as 2500
 neighbor rr-client-2 route-reflector-client
 address-family vpnv4
  neighbor rr-client-2 allow-policy
!
 address-family vpnv6
  neighbor rr-client-2 allow-policy
```

Additional References for Support for iBGP Local-AS

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
Migration of autonomous systems	"BGP Support for Dual AS Configuration for Network AS Migrations" module in the *IP Routing: BGP Configuration Guide, Cisco IOS XE Release 3S*

Technical Assistance

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

Feature Information for BGP—Support for iBGP Local-AS

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 71: Feature Information for BGP—Support for iBGP Local-AS

Feature Name	Releases	Feature Information
BGP—Support for iBGP Local-AS	15.3(2)S 15.3(3)M	Prior to the BGP—Support for Local-AS feature, the **neighbor local-as** command was used on a route reflector to customize AS_PATH attributes for routes received from an eBGP neighbor. The **neighbor local-as** command can now be used to enable the sending of the iBGP attributes (LOCAL_PREF, ORIGINATOR_ID, and CLUSTER_LIST) over an iBGP local-AS session. This functionality is useful when merging two autonomous systems, when it is advantageous to keep the iBGP attributes in routes. Prior to the BGP—Support for iBGP Local-AS feature, the RR should not have been configured to change iBGP attributes. With the introduction of this feature, the RR can be configured to change iBGP attributes, providing more flexibility. The following commands introduced or modified: **neighbor allow-policy**, **neighbor local-as**, **show ip bgp vpnv4**.

CHAPTER 59

BGP-RT and VPN Distinguisher Attribute Rewrite Wildcard

The BGP—RT and VPN Distinguisher Attribute Rewrite Wildcard feature introduces the ability to set a range of route target (RT) community attributes or VPN distinguisher community attributes when mapping them. A network administrator might want to map one or more RTs at an egress ASBR to different RTs at an ingress ASBR. The VPN Distinguisher Attribute feature allows an administrator to map RTs to a VPN distinguisher that is carried through an eBGP and then mapped to RTs at an ingress ASBR. The mapping is achieved by configuring a route map that sets an RT range or VPN distinguisher range of extended community attributes. Specifying a range rather than individual RTs saves time and simplifies the configuration. Furthermore, a VPN distinguisher range allows more than one VPN distinguisher attribute per route-map clause, thereby removing the restriction that applied prior to this feature.

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.

Restrictions for BGP-RT and VPN Distinguisher Attribute Rewrite Wildcard

- A range (specified in the **set extcommunity rt** command or the **set extcommunity vpn-distinguisher** command) can include a maximum of 450 extended communities.
- The VPN distinguisher range is not relayed to an iBGP peer.

Information About BGP-RT and VPN Distinguisher Attribute Rewrite Wildcard

Benefits of RT and VPN Distinguisher Attribute Mapping Range

A network administrator might want to rewrite (or map) one or more route targets (RTs) at an egress ASBR to different RTs at an ingress ASBR. One use case would be to keep the RTs at the egress ASBR private from the ingress ASBR.

The rewrite is achieved by using inbound route maps, matching prefixes to route-map clauses that match inbound RTs, and mapping those RTs to different RTs recognized by the neighbor AS. Such a rewrite configuration could be complex on inbound route maps, with potentially hundreds of RTs that would need to be specified individually (configuring **set extcommunity rt** *value1 value2 value3* ...). If the RTs being attached to the prefixes are consecutive, the configuration can be simplified by specifying a range of RTs. Thus, the benefits of the RT mapping range are saving time and simplifying the configuration.

Likewise, the mapping of RTs to a VPN distinguisher attribute (and vice versa) can also be simplified by specifying a range of RTs or VPN distinguishers. The BGP—VPN Distinguisher Attribute feature allows a network administrator to keep source RTs private from an ASBR in a destination AS. An RT at an egress ASBR is mapped to a VPN distinguisher, the VPN distinguisher is carried through the eBGP, and then it is mapped to an RT at the ingress ASBR.

The RT and VPN Distinguisher Attribute Mapping Range feature introduces the ability to specify a range of either route targets (RTs) or VPN distinguishers when mapping them.

Another benefit applies to setting a VPN distinguisher. Prior to this feature, only one **set extcommunity vpn-distinguisher** value was allowed per route-map clause. With the introduction of the mapping range, a range of VPN distinguishers can be set on a route.

How to Map RTs to RTs Using a Range

Replacing an RT with a Range of RTs

Perform this task on an egress ASBR to replace a route target (RT) with an RT range. Remember to replace the range of RTs with an RT on the ingress ASBR; that task is described in the “Replacing a Range of RTs with an RT” section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip extcommunity-list** *expanded-list* {**permit** | **deny**} **rt** *value*
4. **exit**
5. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
6. **match extcommunity** *extended-community-list-name*
7. **set extcomm-list** *extcommunity-name* **delete**
8. **set extcommunity rt range** *start-value end-value*
9. **exit**
10. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
11. **exit**
12. **router bgp** *as-number*
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **address-family vpnv4**
15. **neighbor** *ip-address* **activate**
16. **neighbor** *ip-address* **route-map** *map-tag* **out**
17. **exit-address-family**

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* {**permit** | **deny**} **rt** *value* **Example:** `Device(config)# ip extcommunity-list 22 permit rt 101:100`	Configures an IP extended community list to configure Virtual Private Network (VPN) route filtering, such that routes with the specified RT are in the extended community list. • This example permits routes having RT 101:100 into the extended community list 22.
Step 4	**exit** **Example:** `Device(config-extcomm-list)# exit`	Exits the configuration mode and enters the next higher configuration mode.
Step 5	**route-map** *map-tag* {**permit** | **deny**} [*sequence-number*] **Example:**	Configures a route map that permits or denies the routes allowed by the subsequent **match** command.

	Command or Action	Purpose
	`Device(config)# route-map rt-mapping permit 10`	• This example permits the routes allowed by the subsequent **match** command.
Step 6	**match extcommunity** *extended-community-list-name* **Example:** `Device(config-route-map)# match extcommunity 22`	Matches on the specified community list. • For this example, routes that match the extended community list 22 (which was configured in Step 3) are subject to the subsequent **set** commands.
Step 7	**set extcomm-list** *extcommunity-name* **delete** **Example:** `Device(config-route-map)# set extcomm-list 22 delete`	Deletes the RT from routes that are in the specified extended community list. • For this example, RTs are deleted from routes that are in extended community list 22.
Step 8	**set extcommunity rt range** *start-value end-value* **Example:** `Device(config-route-map)# set extcommunity rt range 500:1 500:9`	For the routes that are permitted by the route map, sets the specified RT range of extended community attributes, inclusive. • For this example, routes that match extended community 22 have their RT extended community attribute values set to 500:1, 500:2, 500:3, 500:4, 500:5, 500:6, 500:7, 500:8, and 500:9.
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 rt-mapping permit 20`	(Optional) Configures a route map entry that permits routes. • This example configures a route map entry that permits other routes not subject to the RT-to-RT range mapping. If you do not perform this step, all other routes are subject to an implicit deny.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 12	**router bgp** *as-number* **Example:** `Device(config)# router bgp 3000`	Enters router configuration mode and creates a BGP routing process.

	Command or Action	Purpose
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.103.1 remote-as 3000`	Specifies that the neighbor belongs to the autonomous system.
Step 14	**address-family vpnv4** **Example:** `Device(config-router)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address family-specific configurations.
Step 15	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.168.103.1 activate`	Activates the specified neighbor.
Step 16	**neighbor** *ip-address* **route-map** *map-tag* **out** **Example:** `Device(config-router-af)# neighbor 192.168.103.1 route-map rt-mapping out`	Applies the specified outgoing route map to the specified neighbor.
Step 17	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters privileged EXEC mode.

Replacing a Range of RTs with an RT

Perform this task on an ingress ASBR to replace an RT range of attributes with an RT attribute. This task assumes you already configured the egress ASBR to replace the RT with an RT range; that task is described in the “Replacing an RT with a Range of RTs” section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip extcommunity-list** *expanded-list* {**permit** | **deny**} **rt** *reg-exp*
4. **exit**
5. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
6. **match extcommunity** *extended-community-list-name*
7. **set extcomm-list** *extcommunity-name* **delete**
8. **set extcommunity rt** *value* **additive**
9. **exit**

10. **route-map** *map-tag* {**permit** | **deny**} [*sequence-number*]
11. **exit**
12. **router bgp** *as-number*
13. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
14. **address-family vpnv4**
15. **neighbor** *ip-address* **activate**
16. **neighbor** *ip-address* **route-map** *map-tag* **in**
17. **exit-address-family**

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* {**permit** | **deny**} **rt** *reg-exp* **Example:** `Device(config)# ip extcommunity-list 128 permit rt 500:[1-9]`	Configures an IP extended community list to configure Virtual Private Network (VPN) route filtering, such that routes with the specified RT range are in the extended community list. • This example permits routes having RTs in the range 500:1 to 500:9 into the extended community list 128.
Step 4	**exit** **Example:** `Device(config-extcomm-list)# exit`	Exits the configuration mode and enters the next higher configuration mode.
Step 5	**route-map** *map-tag* {**permit** | **deny**} [*sequence-number*] **Example:** `Device(config)# route-map rtmap2 permit 10`	Configures a route map that permits or denies the routes allowed by the subsequent **match** command. • This example permits the routes allowed by the subsequent **match** command.
Step 6	**match extcommunity** *extended-community-list-name* **Example:** `Device(config-route-map)# match extcommunity 128`	Matches on the specified community list. • In this example, routes that match the extended community list 128 (which was configured in Step 3) are subject to the subsequent **set** commands.
Step 7	**set extcomm-list** *extcommunity-name* **delete** **Example:**	Deletes the RTs in the range from routes that are in the specified extended community list.

	Command or Action	Purpose
	`Device(config-route-map)# set extcomm-list 128 delete`	• In this example, RTs in the range are deleted from routes that are in extended community list 128.
Step 8	**set extcommunity rt** *value* **additive** **Example:** `Device(config-route-map)# set extcommunity rt 400:1 additive`	Sets the routes that are permitted by the route map with the specified RT. • In this example, routes that match extended community 128 have their RT set to 400:1. The **additive** keyword causes the RT to be added to the RT list without replacing any RTs.
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 rtmap2 permit 20`	(Optional) Configures a route map entry that permits routes. • This example configures a route map entry that permits other routes not subject to the RT-range-to-RT mapping. If you do not perform this step, all other routes are subject to an implicit deny.
Step 11	**exit** **Example:** `Device(config-route-map)# exit`	Exits route-map configuration mode and enters global configuration mode.
Step 12	**router bgp** *as-number* **Example:** `Device(config)# router bgp 4000`	Enters router configuration mode and creates a BGP routing process.
Step 13	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router)# neighbor 192.168.0.50 remote-as 4000`	Specifies that the neighbor belongs to the autonomous system.
Step 14	**address-family vpnv4** **Example:** `Device(config-router-af)# address-family vpnv4`	Enters address family configuration mode to configure BGP peers to accept address-family-specific configurations.
Step 15	**neighbor** *ip-address* **activate** **Example:**	Activates the specified neighbor.

	Command or Action	Purpose
	`Device(config-router-af)# neighbor 192.168.0.50 activate`	
Step 16	**neighbor** *ip-address* **route-map** *map-tag* **in** **Example:** `Device(config-router-af)# neighbor 192.168.0.50 route-map rtmap2 in`	Applies the specified incoming route map to the specified neighbor.
Step 17	**exit-address-family** **Example:** `Device(config-router-af)# exit-address-family`	Exits address family configuration mode and enters privileged EXEC mode.

Configuration Examples for BGP-RT and VPN Distinguisher Attribute Rewrite Wildcard

Example: Replacing an RT with a Range of RTs

In the following example, on the egress ASBR, routes having RT 101:100 are in the extended community list 22. A route-map named rt-mapping matches on extended community list 22 and deletes the RT from routes in the community list. Routes that match the community list have their RT set to an RT in the range from 500:1 to 500:9. The route map is applied to the neighbor 192.168.103.1.

Egress ASBR

```
ip extcommunity-list 22 permit rt 101:100
!
route-map rt-mapping permit 10
 match extcommunity 22
 set extcomm-list 22 delete
 set extcommunity rt range 500:1 500:9
!
route-map rt-mapping permit 20
!
router bgp 3000
 neighbor 192.168.103.1 remote-as 3000
 address-family vpnv4
  neighbor 192.168.103.1 activate
  neighbor 192.168.103.1 route-map rt-mapping out
  exit-address-family
!
```

On the ingress ASBR, RTs in the range 500:1 to 500:9 belong to extended community list 128. A route map named rtmap2 maps those RTs to RT 400:1. The route map is applied to the neighbor 192.168.0.50.

Ingress ASBR

```
ip extcommunity-list 128 permit RT:500:[1-9]
!
route-map rtmap2 permit 10
 match extcommunity 128
 set extcomm-list 128 delete
 set extcommunity rt 400:1 additive
!
route-map rtmap2 permit 20
!
router bgp 4000
 neighbor 192.168.0.50 remote-as 4000
 address-family vpnv4
  neighbor 192.168.0.50 activate
  neighbor 192.168.0.50 route-map rtmap2 in
  exit-address-family
!
```

Example: Replacing an RT with a Range of VPN Distinguishers

In the following example, on the egress ASBR, routes having RT 201:100 are in the extended community list 22. A route-map named rt-mapping matches on extended community list 22 and deletes the RT from routes in the community list. Routes that match the community list have their VPN distinguishers set to VPN distinguishers in the range from 600:1 to 600:8. The route map is applied to the neighbor 192.168.103.1.

Egress ASBR

```
ip extcommunity-list 22 permit rt 201:100
!
route-map rt-mapping permit 10
 match extcommunity 22
 set extcomm-list 22 delete
 set extcommunity vpn-distinguisher range 600:1 600:8
!
route-map rt-mapping permit 20
!
router bgp 3000
 neighbor 192.168.103.1 remote-as 3000
 address-family vpnv4
  neighbor 192.168.103.1 activate
  neighbor 192.168.103.1 route-map rt-mapping out
  exit-address-family
!
```

On the ingress ASBR, VPN distinguishers in the range 600:1 to 600:8 belong to extended community list 101. A route map named rtmap2 maps those VPN distinguishers to RT range 700:1 700:10. The route map is applied to the neighbor 192.168.0.50. The additive option adds the new range to the existing value without replacing it.

Ingress ASBR

```
ip extcommunity-list 101 permit VD:600:[1-8]
!
route-map rtmap2 permit 10
 match extcommunity 101
```

```
 set extcomm-list 101 delete
 set extcommunity rt 700:1 700:10 additive
!
route-map rtmap2 permit 20
!
router bgp 4000
 neighbor 192.168.0.50 remote-as 4000
 address-family vpnv4
  neighbor 192.168.0.50 activate
  neighbor 192.168.0.50 route-map rtmap2 in
  exit-address-family
!
```

Additional References for BGP-RT and VPN Distinguisher Attribute Rewrite Wildcard

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference
BGP—VPN Distinguisher Attribute	"BGP—VPN Distinguisher Attribute" module in the *IP Routing: BGP Configuration Guide, Cisco IOS XE Release 3S*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP-RT and VPN Distinguisher Attribute Rewrite Wildcard

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 72: Feature Information for BGP—RT and VPN Distinguisher Attribute Rewrite Wildcard

Feature Name	Releases	Feature Information
BGP—RT and VPN Distinguisher Attribute Rewrite Wildcard	15.3(2)S 15.3(3)M	The BGP—RT and VPN Distinguisher Attribute Rewrite Wildcard feature introduces the ability to set a range of route target (RT) community attributes or VPN distinguisher community attributes when mapping them. A network administrator might want to map one or more RTs at an egress ASBR to different RTs at an ingress ASBR. The VPN Distinguisher Attribute feature allows an administrator to map RTs to a VPN distinguisher that is carried through an eBGP and then mapped to RTs at an ingress ASBR. The mapping is achieved by configuring a route map that sets an RT range or VPN distinguisher range of extended community attributes. Specifying a range rather than individual RTs saves time and simplifies the configuration. Furthermore, a VPN distinguisher range allows more than one VPN distinguisher attribute per route-map clause, thereby removing the restriction that applied prior to this feature. The following commands were modified: • **set extcommunity rt** • **set extcommunity vpn-distinguisher**

CHAPTER 60

eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6)

The eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6) feature allows you to configure multipath load sharing among native IPv4 and IPv6 external Border Gateway Protocol (eBGP) and internal BGP (iBGP) paths for improved load balancing in deployments. This module explains the feature and how to configure it.

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 eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6)

eiBGP Multipath for Non-VRF Interfaces Overview

The Border Gateway Protocol (BGP) path-selection algorithm prefers external BGP (eBGP) paths over internal BGP (iBGP) paths. With the eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6) feature, this algorithm is modified to allow multipath load sharing among native IPv4 and IPv6 eBGP and iBGP paths. Prior to the eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6) feature, this functionality was only available on VPN routing and forwarding (VRF) instances. With this feature, the functionality is extended to non-VRF interfaces. The **maximum-paths** command allows you to configure BGP to install multiple paths in the Routing

Information Base (RIB) for multipath load sharing. The BGP best path algorithm selects a single multipath as the best path and advertises the path to BGP peers. Other multipaths are inserted into both the BGP table and the RIB, and these multipaths are used by Cisco Express Forwarding to perform load balancing, which is performed either on a per-packet basis or on a per-source or per-destination basis.

This feature can be configured on a customer provider edge (PE) device. However, the feature should be configured only on one PE device at the customer site. If this feature is configured on more than one PE device, some parts of the traffic may loop between the PE devices at the customer site. Therefore, it is important to set up the feature appropriately to avoid traffic loops. This feature is enabled by default.

How to Configure eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6)

Enabling IPv4/IPv6 Multipaths for Non-VRF Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. Enter one of the following:
    - **address-family ipv4 unicast**
    - **address-family ipv6 unicast**
5. **maximum-paths eibgp** *number*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system-number* **Example:** `Device(config)# router bgp 64496`	Enters router configuration mode to create or configure a Border Gateway Protocol (BGP) routing process.
Step 4	Enter one of the following: • **address-family ipv4 unicast**	Enters IPv4 or IPv6 address family configuration mode.

	Command or Action	Purpose

	• **address-family ipv6 unicast**
	Example:
	`Device(config-router)# address-family ipv4 unicast`
	`Device(config-router)# address-family ipv6 unicast`	

Step 5	**maximum-paths eibgp** *number* **Example:** `Device(config-router-af)# maximum-paths eibgp 3`	Forwards packets over multiple external BGP (eBGP) and internal BGP (iBGP) paths.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6)

Example: Enabling IPv4/IPv6 Multipaths in Non-VRF Interfaces

The following example shows how to enable IPv4 multipaths on non-VRF interfaces.

```
Device> enable
Device# configure terminal
Device(config)# router bgp 64496
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# maximum-paths eibgp 4
Device(config-router-af)# end
```

The following example shows how to enable IPv6 multipaths on non-VRF interfaces.

```
Device> enable
Device# configure terminal
Device(config)# router bgp 64497
Device(config-router)# address-family ipv6 unicast
Device(config-router-af)# maximum-paths eibgp 4
Device(config-router-af)# end
```

Feature Information for eiBGP Multipath for Non-VRF Interfaces (IPv4/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 73: Feature Information for eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6)

Feature Name	Releases	Feature Information
eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6)	15.4(1)T	The eiBGP Multipath for Non-VRF Interfaces (IPv4/IPv6) feature allows you to configure multipath load sharing among native IPv4 and IPv6 external Border Gateway Protocol (eBGP) and internal BGP (iBGP) paths for improved load balancing in deployments. The following command was modified: **maximum-paths eibgp**.

CHAPTER 61

L3VPN iBGP PE-CE

The L3VPN iBGP PE-CE feature enables the provider edge (PE) and customer edge (CE) devices to exchange Border Gateway Protocol (BGP) routing information by peering as iBGP instead of as external BGP peering between the PE and CE.

Finding Feature Information

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

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

Restrictions for L3VPN iBGP PE-CE

We recommend not using the soft-reconfiguration inbound or BGP soft-reconfig-backup feature with the iBGP PE CE.

Information About L3VPN iBGP PE-CE

L3VPN iBGP PE-CE

When BGP is used as the provider edge (PE) or customer edge (CE) routing protocol, the peering sessions are configured as an external peering between the VPN provider autonomous system (AS) and the customer network autonomous system. The L3VPN iBGP PE-CE feature enables the PE and CE devices to exchange Border Gateway Protocol (BGP) routing information by peering as internal Border Gateway Protocol (iBGP) instead of the widely used external BGP peering between the PE and the CE. This mechanism applies at each PE device where a VRF-based CE is configured as iBGP. This eliminates the need for service providers (SPs) to configure autonomous system override for the CE. With this feature enabled, there is no need to configure the virtual private network (VPN) sites using different autonomous systems.

The introduction of the **neighbor internal-vpn-client** command enables PE devices to make an entire VPN cloud act like an internal VPN client to the CE devices. These CE devices are connected internally to the VPN cloud through the iBGP PE-CE connection inside the VRF. After this connection is established, the PE device encapsulates the CE-learned path into an attribute called ATTR_SET and carries it in the iBGP-sourced path throughout the VPN core to the remote PE device. At the remote PE device, this attribute is assigned with individual attributes and the source CE path is extracted and sent to the remote CE devices. ATTR_SET is an optional transitive attribute that carries a set of BGP path attributes. It can include any BGP attribute that can occur in a BGP update message as received from the source CE device.

How to Configure L3VPN iBGP PE-CE

Configuring L3VPN iBGP PE-CE

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4 vrf** *name*
5. **neighbor** *ip-address* **internal-vpn-client**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device(config)# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 100`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family ipv4 vrf** *name* **Example:** `Device(config-router)# address-family ipv4 vrf blue`	Enters address family configuration mode and configures VPN routing and forwarding.
Step 5	**neighbor** *ip-address* **internal-vpn-client** **Example:** `Device(config-router-af)# neighbor 10.0.0.1 internal-vpn-client`	Defines a neighboring device with which to exchange routing information. The **neighbor internal-vpn-client** command stacks the iBGP-CE neighbor path in the VPN attribute set .

Configuration Examples for L3VPN iBGP PE-CE

Example: Configuring L3VPN iBGP PE-CE

The following example shows how to configure L3VPN iBGP PE-CE:

```
Device# enable
Device(config)# configure terminal
Device(config)# router bgp 100
Device(config-router)# address-family ipv4 vrf blue
Device(config-router-af)# neighbor 10.0.0.1 internal-vpn-client
```

Additional References for L3VPN iBGP PE-CE

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 website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for L3VPN iBGP PE-CE

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 74: Feature Information for L3VPN iBGP PE-CE

Feature Name	Releases	Feature Information
L3VPN iBGP PE-CE	15.4(1)T	The L3VPN iBGP PE-CE feature enables the provider edge (PE) and customer edge (CE) devices to exchange Border Gateway Protocol (BGP) routing information by peering as iBGP instead of as external BGP between the PE and CE. The **neighbor internal-vpn-client** command was introduced.

CHAPTER 62

BGP PBB EVPN Route Reflector Support

The BGP PBB EVPN Route Reflector Support feature provides Boarder Gateway Protocol (BGP) route reflector functionality for Ethernet VPN (EVPN) and provider backbone bridging (PBB) EVPN of Layer 2 VPN address family. EVPN enables customer MAC addresses as routable addresses and distributes them in BGP to avoid any data plane MAC address learning over the Multiprotocol Label Switching (MPLS) core network. The route reflector is enhanced to store the received EVPN updates without configuring EVPN explicitly on the route reflector and then advertises these updates to other provider edge (PE) devices so that the PEs do not need to have a full mesh of BGP sessions.

Finding Feature Information

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

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

Prerequisites for BGP PBB EVPN Route Reflector Support

- Before you configure the BGP PBB EVPN Route Reflector Support feature, you must configure the RT filter unicast address family type to support for EVPN address family. For more information, see the "Configuring BGP: RT Constrained Route Distribution" module in the *IP Routing: BGP Configuration Guide*.

- The EVPN Subsequent Address Family Identifier (SAFI) needs to be enabled globally before you enable it under the BGP neighbor.

Information About BGP PBB EVPN Route Reflector Support

EVPN Overview

Ethernet VPN (EVPN) allows Multiprotocol Label Switching (MPLS) networks to provide multipoint Layer 2 VPN (L2VPN) services.

In EVPN, the customer MAC addresses are learned in the data plane over links connecting customer devices (CE) to the provider edge (PE) devices. The MAC addresses are then distributed over the Multiprotocol Label Switching (MPLS) core network using Boarder Gateway Protocol (BGP) with an MPLS label identifying the service instance. A single MPLS label per EVPN instance is sufficient as long as the receiving PE device performs a MAC lookup in the disposition path. Receiving PE devices inject these routable MAC addresses into their Layer 2 routing information base (RIB) and forwarding information base (FIB) along with their associated adjacencies.

EVPN defines a BGP Network Layer Reachability Information (NLRI) that advertises different route types and route attributes. The EVPN NLRI is carried in BGP using BGP multiprotocol extensions with an Address Family Identifier (AFI) and a Subsequent Address Family Identifier (SAFI). BGP drops unsupported route types and does not propagate them to neighbors.

BGP EVPN Autodiscovery Support on Route Reflector

By default, routes received from an internal BGP (iBGP) peer are not sent to another iBGP peer unless a full mesh configuration is formed between all Boarder Gateway Protocol (BGP) devices within an autonomous system (AS). Configuring a route reflector allows a device to advertise or reflect the iBGP learned routes to other iBGP speakers.

Ethernet VPN (EVPN) Autodiscovery supports BGP route reflectors. A BGP route reflector can be used to reflect BGP EVPN prefixes without EVPN being explicitly configured on the route reflector. The route reflector does not participate in autodiscovery; that is, no pseudowires are set up between the route reflector and the provider edge (PE) devices. A route reflector reflects EVPN prefixes to other PE devices so that these PE devices do not need to have a full mesh of BGP sessions. The network administrator configures only the BGP EVPN address family on a route reflector.

BGP uses the Layer 2 VPN (L2VPN) routing information base (RIB) to store endpoint provisioning information, which is updated each time any Layer 2 virtual forwarding instance (VFI) is configured. The prefix and path information is stored in the L2VPN database, which allows BGP to make decisions about the best path. When BGP distributes the endpoint provisioning information in an update message to all its BGP neighbors, this endpoint information is used to configure a pseudowire mesh to support L2VPN-based services.

EVPN Address Family

BGP supports Layer 2 VPN (L2VPN) EVPN address family under router configuration mode to carry L2VPN EVPN autodiscovery and signaling Network Layer Reachability Information (NLRI) to Boarder Gateway Protocol (BGP) neighbors. This address family is allowed on both internal BGP (iBGP) and external BGP (eBGP) neighbors under default virtual routing and forwarding (VRF) for both IPv4 and IPv6 neighbors. The EVPN SAFI is not supported under VRF and VRF neighbors.

How to Configure BGP PBB EVPN Route Reflector Support

Configuring BGP PBB EVPN Route Reflector

Perform this task on the Boarder Gateway Protocol (BGP) route reflector to configure the device as a BGP route reflector and configure the specified neighbor as its client and to display the information from the BGP routing table.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family l2vpn [vpls | evpn]**
5. **neighbor** {*ip-address* | *peer-group-name*} **activate**
6. **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client**
7. **end**
8. **show bgp l2vpn evpn all**
9. **debug bgp l2vpn evpn updates**
10. **clear bgp l2vpn evpn**

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 1`	Configures a BGP routing process and enters router configuration mode.
Step 4	**address-family l2vpn [vpls | evpn]** **Example:** `Device(config-router)# address-family l2vpn evpn`	Specifies the L2VPN address family and enters address family configuration mode. The optional **evpn** keyword specifies that EVPN endpoint provisioning information is to be distributed to BGP peers.
Step 5	**neighbor** {*ip-address* | *peer-group-name*} **activate** **Example:** `Device(config-router-af)# neighbor 10.0.0.2 activate`	Enables PBB EVPN with the specified BGP neighbor.

	Command or Action	Purpose
Step 6	**neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **route-reflector-client** **Example:** `Device(config-router-af)# neighbor 10.0.0.2 route-reflector-client`	Configures the local device as the BGP route reflector and the specified neighbor as one of its clients.
Step 7	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.
Step 8	**show bgp l2vpn evpn all** **Example:** `Device# show bgp l2vpn evpn all`	(Optional) Displays the complete L2VPN EVPN database.
Step 9	**debug bgp l2vpn evpn updates** **Example:** `Device# debug bgp l2vpn evpn updates events`	(Optional) Specifies debugging messages for BGP update.
Step 10	**clear bgp l2vpn evpn** **Example:** `Device# clear bgp l2vpn evpn *`	(Optional) Specifies that all current BGP sessions will be reset.

Configuration Examples for BGP PBB EVPN Route Reflector Support

Example: Configuring BGP PBB EVPN Route Reflector

In the following example, the local device is a route reflector. It passes learned iBGP routes to the neighbor at 10.0.0.2:

```
Device# configure terminal
Device(config)# router bgp 1
Device(config-router)# address-family l2vpn evpn
Device(config-router-af)# neighbor 10.0.0.2 activate
Device(config-router-af)# neighbor 10.0.0.2 route-reflector-client
Device(config-router-af)# exit address-family
```

In the following example, the **show bgp l2vpn evpn all route-type 1** command displays the Ethernet autodiscovery route information:

```
show bgp l2vpn evpn all route-type 1

BGP routing table entry for
[1][100.100.100.100:11111][AAAABBBBCCCCDDDDEEEE][23456789][101234]/25, version 2
Paths: (1 available, best #1, table EVPN-BGP-Table)
  Advertised to update-groups:
     1          2          3
```

```
Refresh Epoch 1
Local, (Received from a RR-client)
  19.0.101.1 from 19.0.101.1 (19.0.101.1)
    Origin IGP, localpref 100, valid, internal, best
    Extended Community: RT:100:101 EVPN LABEL:0x1:Label-101234
    rx pathid: 0, tx pathid: 0x0
```

Additional References for BGP PBB EVPN Route Reflector Support

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 4456	*BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*
RFC 4684	*Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP PBB EVPN Route Reflector Support

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 75: Feature Information for BGP PBB EVPN Route Reflector Support

Feature Name	Releases	Feature Information
BGP PBB EVPN Route Reflector Support	15.4(2)T	The BGP PBB EVPN Route Reflector Support feature provides Boarder Gateway Protocol (BGP) route reflector functionality for Ethernet VPN (EVPN) and provider backbone bridging (PBB) EVPN of Layer 2 VPN address family. EVPN enables customer MAC addresses as routable addresses and distributes them in BGP to avoid any data plane MAC address learning over the Multiprotocol Label Switching (MPLS) core network. The route reflector is enhanced to store the received EVPN updates without configuring EVPN explicitly on the route reflector and then advertises these updates to other provider edge (PE) devices so that the PEs do not need to have a full mesh of BGP sessions. The following command was modified: **address-family l2vpn**.

CHAPTER 63

BGP-RTC for Legacy PE

The BGP-Route Target Constrain (RTC) for Legacy PE feature helps to prevent the propagation of VPN Network Layer Reachability Information (NLRI) to a provider edge (PE) device that is not interested in the VPN. This feature builds an outbound filter used by a Boarder Gateway Protocol (BGP) speaker to decide which routes to pass to its peer and propagates route target (RT) reachability information between internal BGP (iBGP) meshes.

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.

Prerequisites for BGP-RTC for Legacy PE

Before you configure the BGP-RTC for Legacy PE feature, you must configure the RT filter unicast address family type. For more information, see "Configuring BGP: RT Constrained Route Distribution" module in the *IP Routing: BGP Configuration Guide*.

Information About BGP-RTC for Legacy PE

Overview of BGP-RTC for Legacy PE

The BGP—RTC for Legacy PE feature makes use of VPN unicast route exchange from the legacy provider edge (PE) devices to a new Boarder Gateway Protocol (BGP) speaker (route reflector [RR]) to signal route target (RT) membership. The legacy PEs announce a set of special routes with mapped RTs to the RR along with a standard community. The presence of the community triggers the RR to extract the RTs and build RT membership information.

In scenarios where VPN membership is normal, this functionality helps reduce the scaling requirements on the PE devices and the RRs. The PE devices need not to spend resources for filtering out unwanted routes. The BGP peers that have common outbound policies are grouped under a single format group. Separate replication groups are used within a format group to separate BGP peers with its own peer-based policies. The Route Target Constrain (RTC)-capable peers are placed in separate format groups. Each RTC peers have a separate replication group. When legacy RT is configured for a peer, then it must be treated the same way as the RTC peer except that there is no capability negotiation.

Legacy PE Support-PE Behavior

Each legacy Route Target Constrain (RTC) speaking neighbor is assigned a separate replication group. BGP checks the VPN table for any route with a reserved community value and uses it to create RTC network from the VPN prefix received from a legacy RTC peer with community values. The PE device uses the existing VPN advertisement mechanism to convey route target (RT) membership from the legacy provider edge (PE) devices. The route reflector (RR) processes advertisement mechanisms of RT membership information from legacy PE devices. RRs translate the legacy PE RT membership information to equivalent RTC Network Layer Reachability Information (NLRIs) to propagate to other RRs.

Legacy PE Support-RR Behavior

Route reflectors (RR) identify routes from legacy provider edge (PE) devices for retrieving route target (RT) membership information by the community value and filter VPN routes to legacy PE devices. RRs use the existing VPN advertisement mechanism to convey and process RT membership from the legacy PEs. The legacy PE RT membership information is translated into equivalent RT membership Network Layer Reachability Information (NLRI) from the client to propagate to other RRs. The RR then creates the route target filter list for each legacy client by collecting the entire set of route targets.

How to Configure BGP-RTC for Legacy PE

Configuring BGP-RTC for Legacy PE

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **router bgp** *as-number*
4. **address-family** {**vpnv4** | **vpnv6** } **unicast**
5. **neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **accept-route-legacy-rt**
6. **address-family rtfilter**
7. **end**
8. **show ip bgp vpnv4 all update-group** *update-group*
9. **show ip bgp vpnv4 all neighbors** {*ip-address* | *ipv6-address*}
10. **show ip bgp vpnv4 all peer-group**
11. **debug ip bgp all updates in**

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 1`	Configures a Boarder Gateway Protocol (BGP) routing process and enters router configuration mode.
Step 4	**address-family** {**vpnv4** | **vpnv6** } **unicast** **Example:** `Device(config-router)# address-family vpnv4 unicast`	Specifies the VPNv4 or VPNv6 address family and enters address family configuration mode.
Step 5	**neighbor** {*ip-address* | *peer-group-name* | *ipv6-address*} **accept-route-legacy-rt** **Example:** `Device(config-router-af)# neighbor 10.0.0.1 accept-route-legacy-rt`	Configures the neighbor on the route reflector (RR) to treat the provider edge (PE) device as a legacy PE for the route target (RT) and accepts VPN routes tagged with the special community.
Step 6	**address-family rtfilter** **Example:** `Device(config-router-af)# address-family rtfilter`	Specifies the RT filter address family type.
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 vpnv4 all update-group** *update-group* **Example:**	(Optional) Displays the information about neighbors in the update group.

	Command or Action	Purpose
	`Device# show ip bgp vpnv4 all update-group 2`	
Step 9	**show ip bgp vpnv4 all neighbors** {*ip-address* | *ipv6-address*} **Example:** `Device# show ip bgp vpnv4 all neighbors 192.168.3.3`	(Optional) Displays information about the BGP VPNv4 neighbor.
Step 10	**show ip bgp vpnv4 all peer-group** **Example:** `Device# show ip bgp vpnv4  all peer-group`	(Optional) Displays information about the peer groups.
Step 11	**debug ip bgp all updates in** **Example:** `Device# debug ip bgp all updates in`	(Optional) Displays BGP update messages.

Configuration Examples for BGP-RTC for Legacy PE

Example: BGP-RTC for Legacy PE

Configuration on the Route Reflector

The following example shows how to configure the neighbor on the route reflector (RR) to treat the provider edge (PE) device as a legacy PE for the route target (RT) and accept VPN routes tagged with the special community:

```
Device# configure terminal
Device(config)# router bgp 1
Device(config-router)# address-family vpnv4 unicast
Device(config-router-af)# neighbor 10.1.1.1 accept-route-legacy-rt
Device(config-router-af)# address-family rtfilter
Device(config-router-af)# exit address-family
```

Configuration on the Legacy PE

The following example shows how to create a route filter VRF and attach an export map that collects and carries all RTs locally configured on Layer 3 VPN virtual routing and forwarding (VRF):

```
ip vrf route-filter
 rd 55:1111
 export map SET_RT

route-map SET_RT permit 10
 match ip address prefix-list RT_NET1
 set community 4294901762 (0xFFFF0002)
 set extcommunity rt  255.220.0.0:12241 255.220.0.0:12242 additive
 set extcommunity rt  255.220.0.0:12243 255.220.0.0:12244 additive
 set extcommunity rt  255.220.0.0:12245 255.220.0.0:12246 additive
 set extcommunity rt  255.220.0.0:12247 255.220.0.0:12248 additive
 set extcommunity rt  255.220.0.0:12249 255.220.0.0:12250 additive
```

```
!
route-map SET_RT permit 20
 match ip address prefix-list RT_NET2
 set community 4294901762 (0xFFFF0002)
 set extcommunity rt  255.220.0.0:12251 255.220.0.0:12252 additive
 set extcommunity rt  255.220.0.0:12253 255.220.0.0:12254 additive
 set extcommunity rt  255.220.0.0:12255 additive
!

ip route vrf route-filter 5.5.5.5 255.255.255.255 Null0 – (matching prefix-set RT_NET1)
ip route vrf route-filter 6.6.6.6 255.255.255.255 Null0 –(matching prefix-set RT_NET2)

route-map LEG_PE permit 10
 match ip address prefix-list RT_NET1 RT_NET2
 set community no-advertise additive
```

The following example shows how to apply the route map to a VPNv4 neighbor:

```
router bgp 55
 address-family vpnv4 unicast
 neighbor x.x.x.x  route-map LEG_PE out
```

The following example shows how to source a static route into a Boarder Gateway Protocol (BGP) network using a network statement:

```
router bgp 55
 address-family ipv4 vrf route-filter
 network 5.5.5.5 mask 255.255.255.255
 network 6.6.6.6 mask 255.255.255.255
```

Additional References for BGP-RTC for Legacy PE

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
Configuring BGP: RT Constrained Route Distribution	"Configuring BGP: RT Constrained Route Distribution" module in the *IP Routing: BGP Configuration Guide*

Standards and RFCs

Standard/RFC	Title
RFC 4684	*Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP-RTC for Legacy PE

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 76: Feature Information for BGP-RTC for Legacy PE

Feature Name	Releases	Feature Information
BGP-RTC for Legacy PE	15.4(2)T	The BGP-RTC for Legacy PE feature helps to prevent the propagation of VPN Network Layer Reachability Information (NLRI) to a provider edge (PE) device that is not interested in the VPN. This feature builds an outbound filter used by a Boarder Gateway Protocol (BGP) speaker to decide which routes to pass to its peer and propagates route target (RT) reachability information between internal BGP (iBGP) meshes. The **neighbor accept-route-legacy-rt** command was introduced.

CHAPTER 64

BGP Monitoring Protocol

The BGP Monitoring Protocol (BMP) feature supports the following functionality to monitor Border Gateway Protocol (BGP) neighbors, also called BMP clients:

- Configure devices to function as BMP servers, and set up parameters on the servers, that are required for monitoring of the BGP neighbors.
- Establish connectivity of the BMP servers with BGP neighbors for monitoring.
- Generate statistics report from monitoring the BGP neighbors.
- Perform appropriate error handling on the BGP neighbors.
- Graceful scale up and degradation to the point of closing connectivity between the BMP servers and BGP neighbors.

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.

Prerequisites for BGP Monitoring Protocol

Before you configure BGP Monitoring Protocol (BMP) servers, you must configure Border Gateway Protocol (BGP) neighbors, which function as BMP clients, and establish a session with its peers using either IPv4/IPv6 or VPNv4/VPNv6 address-family identifiers.

Information About BGP Monitoring Protocol

BGP Monitoring Protocol Overview

The BGP Monitoring Protocol (BMP) feature enables monitoring of BGP neighbors (called BMP clients). You can configure a device to function as a BMP server, which monitors either one or several BMP clients, which in turn, has several active peer sessions configured. You can also configure a BMP client to connect to one or more BMP servers. The BMP feature enables configuration of multiple BMP servers (configured as primary servers) to function actively and independent of each other, simultaneously to monitor BMP clients.

Each BMP server is specified by a number and you can use command-line interface (CLI) to configure parameters such as IP address, port number, and so on. Upon activation of a BMP server, it attempts to connect to BMP clients by sending an initiation message. The CLI enables multiple—independent and asynchronous—BMP server connections.

BGP neighbors, called BMP clients, are configured to send data to specific BMP servers for monitoring purposes. These clients are configured in a queue. When a request for a connection arrives from BMP clients to BMP servers, the connection is established based on the order in which the requests arrived. Once the BMP server connects with the first BMP neighbor, it sends out refresh requests to monitor the BMP clients and starts monitoring those BMP clients with whom the connection is already established.

The session connection requests from the other BMP clients in queue to the BMP servers initiates after an initial delay that you can configure using the **initial-delay** command. If a connection establishes but fails later, due to some reason, the connection request is retried after a delay, which you can configure using **failure-retry-delay** command. If there is repeated failure in connection establishment, the connection retries are delayed based on the delay configured using the **flapping-delay** command. Configuring the delay for such requests becomes significant because the route refresh requests that are sent to all connected BMP clients causes considerable network traffic and load on the device.

To avoid excessive load on the device, the BMP servers sends route refresh requests to individual BMP clients at a time, in the order in which connections are established in the queue. Once a BMP client that is already connected is in the "reporting" state, it sends a "peer-up" message to the BMP server. After the client receives a route-refresh request, route monitoring begins for that neighbor. Once the route refresh request ends, the next neighbor in the queue is processed. This cycle continues until all "reporting" BGP neighbors are reported and all routes sent by these "reporting" BGP neighbors are continuously monitored. If a neighbor establishes after BMP monitoring has begun, it does not require a route-refresh request. All received routes from that client is sent to BMP servers.

It is advantageous to batch up refresh requests from BMP clients, if several BMP servers are activated in quick succession. Use the **bmp initial-refresh delay** command to configure a delay in triggering the refresh mechanism when the first BMP server comes up. If other BMP servers come online within this time-frame, only one set of refresh requests is sent to the BMP clients. You can also configure the **bmp initial-refresh skip** command to skip all refresh requests from BMP servers and just monitor all incoming messages from the peers.

In a client-server configuration, it is recommended that the resource load of the devices be kept minimal and adding excessive network traffic must be avoided. In the BMP configuration, you can configure various delay timers on the BMP server to avoid flapping during connection between the server and client. To avoid excessive message throughput or high usage of system resources, you can configure the maximum buffer limit for the BMP session.

How to Configure BGP Monitoring Protocol

Configuring a BGP Monitoring Protocol Session

Perform this task to configure BGP Monitoring Protocol (BMP) session parameters for the BMP servers to establish connectivity with BMP clients.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **bmp** {**buffer-size** *buffer-bytes* | **initial-refresh** {**delay** *refresh-delay* | **skip**} | **server** *server-number-n*
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** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**bmp** {**buffer-size** *buffer-bytes*	**initial-refresh** {**delay** *refresh-delay*	**skip**}	**server** *server-number-n* **Example:** `Device(config-router)# bmp initial-refresh delay 30`	Configures BMP parameters for BGP neighbors and enters BMP server configuration mode to configure BMP servers.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.

Configuring BGP Monitoring Protocol on BGP Neighbors

Perform this task to activate BGP Monitoring Protocol (BMP) on BGP neighbors (also called BMP clients) so that the client activity is monitored by the BMP server configured on the neighbor.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **neighbor** {*ipv4-addr* | *neighbor-tag* | *ipv6-addr*} **bmp-activate** {**all** | **server** *server-number-1* [**server** *server-number-2* . . . [**server** *server-number-n*]]}
   - Repeat Steps 1 to 4 to configure other BMP clients in the session.
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** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**neighbor** {*ipv4-addr* | *neighbor-tag* | *ipv6-addr*} **bmp-activate** {**all** | **server** *server-number-1* [**server** *server-number-2* . . . [**server** *server-number-n*]]} • Repeat Steps 1 to 4 to configure other BMP clients in the session. **Example:**	Activates BMP monitoring on a BGP neighbor.

	Command or Action	Purpose
	`Device(config-router)# neighbor 30.1.1.1 bmp-activate server 1 server 2`	
Step 5	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.

Configuring BGP Monitoring Protocol Servers

Perform this task to configure BGP Monitoring Protocol (BMP) servers and its parameters in BMP server configuration mode.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **bmp** {**buffer-size** *buffer-bytes* | **initial-refresh** {**delay** *refresh-delay* | **skip**} | **server** *server-number-n*
5. **activate**
6. **address** {*ipv4-addr* | *ipv6-addr*} **port-number** *port-number*
7. **description** **LINE** *server-description*
8. **failure-retry-delay** *failure-retry-delay*
9. **flapping-delay** *flap-delay*
10. **initial-delay** *initial-delay-time*
11. **set ip dscp** *dscp-value*
12. **stats-reporting-period** *report-period*
13. **update-source** *interface-type* *interface-number*
14. **exit-bmp-server-mode**
    - Repeat Steps 1 to 14 to configure other BMP servers in the session.
15. **end**

DETAILED STEPS

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

	Command or Action	Purpose
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**bmp** {**buffer-size** *buffer-bytes* | **initial-refresh** {**delay** *refresh-delay* | **skip**} | **server** *server-number-n* **Example:** `Device(config-router)# bmp server 1`	Enters BMP server configuration mode to configure BMP servers.
Step 5	**activate** **Example:** `Device(config-router-bmpsrvr)# activate`	Initiates a connection between BMP server and BGP neighbors.
Step 6	**address** {*ipv4-addr* | *ipv6-addr*} **port-number** *port-number* **Example:** `Device(config-router-bmpsrvr)# address 10.1.1.1 port-number 8000`	Configures IP address and port number to a specific BMP server.
Step 7	**description LINE** *server-description* **Example:** `Device(config-router-bmpsrvr)# description LINE SERVER1`	Configures a textual description of a BMP server.
Step 8	**failure-retry-delay** *failure-retry-delay* **Example:** `Device(config-router-bmpsrvr)# failure-retry-delay 40`	Configures delay in the retry requests during failures when sending BMP server updates.
Step 9	**flapping-delay** *flap-delay* **Example:** `Device(config-router-bmpsrvr)# flapping-delay 120`	Configures delays in flapping when sending BMP server updates.
Step 10	**initial-delay** *initial-delay-time* **Example:** `Device(config-router-bmpsrvr)# initial-delay 20`	Configures delays in sending initial requests for updates from the BMP servers.
Step 11	**set ip dscp** *dscp-value* **Example:**	Configures the IP Differentiated Services Code Point (DSCP) values for BMP servers.

	Command or Action	Purpose
	`Device(config-router-bmpsrvr)# set ip dscp 5`	
Step 12	**stats-reporting-period** *report-period* **Example:** `Device(config-router-bmpsrvr)# stats-reporting-period 30`	Configures the time interval in which the BMP server receives the statistics report from BGP neighbors.
Step 13	**update-source** *interface-type interface-number* **Example:** `Device(config-router-bmpsrvr)# update-source ethernet 0/0`	Configures the interface source for routing updates on the BMP servers.
Step 14	**exit-bmp-server-mode** • Repeat Steps 1 to 14 to configure other BMP servers in the session. **Example:** `Device(config-router-bmpsrvr)# exit-bmp-server-mode`	Exits from BMP server configuration mode and returns to router configuration mode.
Step 15	**end** **Example:** `Device(config-router)# end`	Returns to privileged EXEC mode.

Verifying BGP Monitoring Protocol

Perform the following steps to verify the configuration for the BGP Monitoring Protocol (BMP) servers and BMP clients:

SUMMARY STEPS

1. **enable**
2. **show ip bgp bmp**
3. **show running-config**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**show ip bgp bmp** **Example:** `Device# show ip bgp bmp neighbors`	Displays information about BMP servers and neighbors.
Step 3	**show running-config** **Example:** `Device# show running-config | section bmp`	Displays information about BMP servers and neighbors.

Monitoring BGP Monitoring Protocol

Perform the following steps to enable debugging and monitor the BGP Monitoring Protocol (BMP) servers.

SUMMARY STEPS

1. **enable**
2. **debug ip bgp bmp**
3. **show debugging**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**debug ip bgp bmp** **Example:** `Device# debug ip bgp bmp server`	Enables debugging of the BMP attributes.
Step 3	**show debugging** **Example:** `Device# show debugging`	Displays information about the types of debugging that are enabled on a device.

Configuration Examples for BGP Monitoring Protocol

Examples for Configuring, Verifying, and Monitoring BGP Monitoring Protocol

Examples: Configuring BGP Monitoring Protocol

Note There are two levels of configuration required for the BGP Monitoring Protocol (BMP) to function as designed. You must enable BMP monitoring on each BGP neighbor (also called BMP client) to which several peers are connected in a network, and establish connectivity between the BMP servers and clients. Then, configure each BMP server in BMP server configuration mode for a specific server with the parameters required for monitoring the associated BMP clients.

The following example shows how to activate BMP on a neighbor with IP address 30.1.1.1, which is monitored by BMP servers (in this case, server 1 and 2):

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# neighbor 30.1.1.1 bmp-activate server 1 server 2
Device(config-router)# end
```

The following example shows how to configure initial refresh delay of 30 seconds for BGP neighbors on which BMP is activated using the **neighbor bmp-activate** command:

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# bmp initial-refresh delay 30
Device(config-router)# bmp buffer-size 2048
Device(config-router)# end
```

The following example show how to enter BMP server configuration mode and initiate connection between a specific BMP server with the BGP BMP neighbors. In this example, connection to clients is initiated from BMP servers 1 and 2 along with configuration of the monitoring parameters:

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# bmp server 1
Device(config-router-bmpsrvr)# activate
Device(config-router-bmpsrvr)# address 10.1.1.1 port-number 8000
Device(config-router-bmpsrvr)# description LINE SERVER1
Device(config-router-bmpsrvr)# failure-retry-delay 40
Device(config-router-bmpsrvr)# flapping-delay 120
Device(config-router-bmpsrvr)# initial-delay 20
Device(config-router-bmpsrvr)# set ip dscp 5
Device(config-router-bmpsrvr)# stats-reporting-period 30
Device(config-router-bmpsrvr)# update-source ethernet 0/0
Device(config-router-bmpsrvr)# exit-bmp-server-mode
Device(config-router)# bmp server 2
```

```
Device(config-router-bmpsrvr)# activate
Device(config-router-bmpsrvr)# address 20.1.1.1 port-number 9000
Device(config-router-bmpsrvr)# description LINE SERVER2
Device(config-router-bmpsrvr)# failure-retry-delay 40
Device(config-router-bmpsrvr)# flapping-delay 120
Device(config-router-bmpsrvr)# initial-delay 20
Device(config-router-bmpsrvr)# set ip dscp 7
Device(config-router-bmpsrvr)# stats-reporting-period 30
Device(config-router-bmpsrvr)# update-source ethernet 2/0
Device(config-router-bmpsrvr)# exit-bmp-server-mode
Device(config-router)# end
```

Examples: Verifying BGP Monitoring Protocol

The following is sample output from the **show ip bgp bmp server** command for server number 1. The attributes displayed are configured in the BMP server configuration mode:

```
Device# show ip bgp bmp server 1

Print detailed info for 1 server number 1.

bmp server 1
address: 10.1.1.1    port 8000
description SERVER1
up time 00:06:22
session-startup route-refresh
initial-delay 20
failure-retry-delay 40
flapping-delay 120
activated
```

The following is sample output from the **show ip bgp bmp server** command for server number 2. The attributes displayed are configured in the BMP server configuration mode:

```
Device# show ip bgp bmp server 2

Print detailed info for 1 server number 2.

bmp server 2
address: 20.1.1.1    port 9000
description SERVER2
up time 00:06:23
session-startup route-refresh
initial-delay 20
failure-retry-delay 40
flapping-delay 120
activated
```

The following is sample output from the **show ip bgp bmp server summary** command after deactivating the BMP server 1 and 2 connections:

```
Device# show ip bgp bmp server summary

Number of BMP servers configured: 2
Number of BMP neighbors configured: 10
Number of neighbors on TransitionQ: 0, MonitoringQ: 0, ConfigQ: 0
Number of BMP servers on StatsQ: 0
BMP Refresh not in progress, refresh not scheduled
Initial Refresh Delay configured, refresh value 30s
BMP buffer size configured, buffer size 2048 MB, buffer size bytes used 0 MB
```

```
ID Host/Net          Port  TCB                 Status  Uptime    MsgSent   LastStat
1  10.1.1.1          8000  0x0                 Down              0
2  20.1.1.1          9000  0x0                 Down              0
```

The following is sample output from the **show ip bgp bmp neighbors** command, which shows the status of the BGP BMP neighbors after reactivating the BMP server 1 and 2 connections:

```
Device# show ip bgp bmp server neighbors

Number of BMP neighbors configured: 10
BMP Refresh not in progress, refresh not scheduled
Initial Refresh Delay configured, refresh value 30s
BMP buffer size configured, buffer size 2048 MB, buffer size bytes used 0 MB

Neighbor           PriQ      MsgQ      CfgSvr#             ActSvr#             RM Sent
30.1.1.1           0         0         1  2                1  2                16
2001:DB8::2001     0         0         1  2                1  2                15
40.1.1.1           0         0         1  2                1  2                26
2001:DB8::2002     0         0         1  2                1  2                15
50.1.1.1           0         0         1  2                1  2                16
60.1.1.1           0         0         1  2                1  2                26
2001:DB8::2002     0         0         1                   1                   9
70.1.1.1           0         0         2                   2                   12
Neighbor           PriQ      MsgQ      CfgSvr#             ActSvr#             RM Sent
80.1.1.1           0         0         1                   1                   10
2001:DB8::2002     0         0         1  2                1  2                16
```

The following is sample output from the **show ip bgp bmp server** command for BMP server number 1 and 2. The statistics reporting interval on BMP server 1 and 2 has been set to 30 seconds, therefore each server receives statistics messages from its connected BGP BMP neighbor in each cycle of 30 seconds:

```
Device# show ip bgp bmp server summary

Number of BMP servers configured: 2
Number of BMP neighbors configured: 10
Number of neighbors on TransitionQ: 0, MonitoringQ: 0, ConfigQ: 0
Number of BMP servers on StatsQ: 0
BMP Refresh not in progress, refresh not scheduled
Initial Refresh Delay configured, refresh value 30s
BMP buffer size configured, buffer size 2048 MB, buffer size bytes used 0 MB

ID Host/Net          Port  TCB                 Status  Uptime    MsgSent   LastStat
1  10.1.1.1          8000  0x2A98B07138        Up      00:38:49  162       00:00:09
2  20.1.1.1          9000  0x2A98E17C88        Up      00:38:49  46        00:00:04


Device# show ip bgp bmp server summary

Number of BMP servers configured: 2
Number of BMP neighbors configured: 10
Number of neighbors on TransitionQ: 0, MonitoringQ: 0, ConfigQ: 0
Number of BMP servers on StatsQ: 0
BMP Refresh not in progress, refresh not scheduled
Initial Refresh Delay configured, refresh value 30s
BMP buffer size configured, buffer size 2048 MB, buffer size bytes used 0 MB

ID Host/Net          Port  TCB                 Status  Uptime    MsgSent   LastStat
1  10.1.1.1          8000  0x2A98B07138        Up      00:40:19  189       00:00:07
2  20.1.1.1          9000  0x2A98E17C88        Up      00:40:19  55        00:00:02
```

Note

If we configure several BGP BMP neighbors to be monitored by the BMP servers, for example 10, then 10 statistics messages are received by both servers in each periodic cycle that is configured.

The following is sample output from the **show running-config** command, which shows the running configuration on the device:

```
Device# show running-config | section bmp

bmp server 1
address 10.1.1.1 port-number 8000
description SERVER1
initial-delay 20
failure-retry-delay 40
flapping-delay 120
update-source Ethernet0/0
set ip dscp 3
activate
exit-bmp-server-mode
bmp server 2
address 20.1.1.1 port-number 9000
description SERVER2
initial-delay 20
failure-retry-delay 40
flapping-delay 120
update-source Ethernet2/0
set ip dscp 5
activate
exit-bmp-server-mode
bmp initial-refresh delay 30
bmp-activate all
```

Examples: Monitoring BGP Monitoring Protocol

The following example shows how to enable debugging of the various BMP attributes:

```
Device# debug ip bgp bmp event

BGP BMP events debugging is on

Device# debug ip bgp bmp neighbor

BGP BMP neighbor debugging is on

Device# debug ip bgp bmp server

BGP BMP server debugging is on
```

The following is sample output from the **show debugging** command after you enable the BGP BMP server debugging:

```
Device# show debugging

IP routing:
BGP BMP server debugging is on

Device#
```

```
*Apr  8 21:04:13.164: BGPBMP: BMP server connection attempt timer expired for server 1 -
10.1.1.1/8000
*Apr  8 21:04:13.165: BGPBMP: BMP server 1 active open process success - 10.1.1.1/8000
*Apr  8 21:04:13.165: BGPBMP: TCP KA interval is set to 15

Device#

*Apr  8 21:04:15.171: BGPBMP: Register read/write notification callbacks with BMP server 1
 TCB - 10.1.1.1/8000
*Apr  8 21:04:15.171: BGPBMP: Initiation msg sent to BMP server 1 - 10.1.1.1/8000
*Apr  8 21:04:15.171: BGPBMP: BMP server 1 connection - 10.1.1.1/8000 up, invoke refresh
event

Device#

*Apr  8 21:04:16.249: BGPBMP: BMP server connection attempt timer expired for server 2 -
20.1.1.1/9000
*Apr  8 21:04:16.249: BGPBMP: BMP server 2 active open process success - 20.1.1.1/9000
*Apr  8 21:04:16.249: BGPBMP: TCP KA interval is set to 15
*Apr  8 21:04:16.250: BGPBMP: Register read/write notification callbacks with BMP server 2
 TCB - 20.1.1.1/9000
*Apr  8 21:04:16.250: BGPBMP: Initiation msg sent to BMP server 2 - 20.1.1.1/9000
*Apr  8 21:04:16.250: BGPBMP: BMP server 2 connection - 20.1.1.1/9000 up, invoke refresh
event
```

Additional References for BGP Monitoring Protocol

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 website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP Monitoring Protocol

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 77: Feature Information for BGP Monitoring Protocol

Feature Name	Releases	Feature Description
BGP Monitoring Protocol	15.4(2)T	

Feature Name	Releases	Feature Description
		The BMP feature supports the following functionality to enable monitoring of the Border Gateway Protocol (BGP) neighbors, which become BMP clients: • Configure devices to function as BMP servers, and set up parameters on the servers, that are required for monitoring of the BGP neighbors. • Establish connectivity of the BMP servers with BGP neighbors for monitoring. • Generate statistics report from monitoring the BGP neighbors. • Perform appropriate error handling on the BGP neighbors. • Graceful scale up and degradation to the point of closing connectivity between the BMP servers and BGP neighbors. The following commands were introduced or modified: **bmp** **debug ip bgp bmp** **neighbor bmp-activate** **show ip bgp bmp** The following commands were introduced in BMP server configuration mode, to configure specific BMP servers: **activate** **address** **default** **description** **exit-bmp-server-mode** **failure-retry-delay** **flapping-delay**

Feature Name	Releases	Feature Description
		initial-delay **set ip dscp** **stats-reporting-period** **update-source**

CHAPTER 65

VRF Aware BGP Translate-Update

The VRF aware BGP translate-update feature enables multicast forwarding on those customer-edge (CE) devices, which have an older version of Cisco software that does not support multicast BGP (mBGP) routing.

The provider-edge (PE) devices establish a virtual routing and forwarding (VRF) session with the neighbor CE devices, and configure the translate-update feature under an IPv4/IPv6 VRF address family. The PE devices translate the updates from unicast to multicast on CE devices and put them as multicast updates in the Border Gateway Protocol (BGP) VRF routing table of the PE devices for processing.

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.

Prerequisites for VRF Aware BGP Translate-Update

- The VRF aware translate-update feature applies only to IPv4/IPv6 virtual routing and forwarding (VRF) address-families.
- You must use peer-group for the configuration of the neighbor under IPv4/IPv6 VRF address families.
- BGP neighbors that are only capable of unicast routing, must be activated under both unicast and multicast address families.

- BGP neighbors must also be enabled under the compatible multicast address family for the VRF aware translate-update feature to function as designed.
- The provider-edge (PE) devices must have multicast VRF enabled and must have a session established with the customer-edge (CE) devices.

Restrictions for VRF Aware BGP Translate-Update

- You must not configure (nonVRF) IPv4/IPv6 address families for the VRF aware BGP translate-update feature. The IPv4/IPv6 address family must be configured for multicast routing using the Subsequent Address Family Identifier (SAFI) feature.
- The VRF aware BGP translate-update feature does not support configuration of BGP neighbor using peer-template.

Information About VRF Aware BGP Translate-Update

VRF Aware BGP Translate-Update Overview

The VRF aware BGP translate-update feature enables multicast forwarding on those customer-edge (CE) devices, which have an older version of Cisco software that does not support multicast BGP (mBGP) routing.

This feature is analogous to the Subsequent Address Family Identifier (SAFI), which provides the capability to support multicast routing in the service provider's core IPv4 network, but is limited in support to IPv4/IPv6 address families. In the case of the virtual routing and forwarding (VRF) aware BGP translate-update feature, provider-edge (PE) devices establish a VRF session with the neighbor CE devices, and have the translate-update feature configured under an IPv4/IPv6 VRF address family.

When the **neighbor translate-update** command is configured on a PE device under the (IPv4 VRF) address-family configuration mode or the (IPv6 VRF) address-family configuration mode, the PE devices translate the updates from unicast to multicast on CE devices and put them in the Border Gateway Protocol (BGP) VRF routing table of the PE devices, as multicast updates, for processing. If you also configure the optional keyword **unicast**, the updates that are not translated, are placed in the PE device's unicast queue and populates the unicast VRF BGP table. The translation from unicast to multicast routes occurs from CE devices to PE devices only, and the multicast and unicast prefixes are only advertised from the CE device to the PE device's multicast neighbors.

For example, when you configure the VRF aware BGP translate-update feature under a VRF (v1) for a neighbor CE device (CE1), a neighbor topology under the IPv4-multicast-VRF or IPv6-multicast-VRF address-family is added to CE1's session with a PE device (PE1). The multicast-VRF neighbor topology does not actively participate in these multicast sessions and only forwards announcements that arrive from CE1. Once such announcements arrive, they are translated into multicast and placed in the nonactive multicast VRF neighbor's routing table. The Cisco software ensures that the routes advertised by CE1 configured under the IPv4/IPv6 VRF address-family are available on PE1's IPv4/IPv6 multicast VRF v1 address-family BGP table. These routes, along with PE1's IPv4/IPv6 multicast VRF v1 address-family BGP table, are advertised to PE1's multicast peers if you have configured the **neighbor translate-update** command . The routes are also advertised to PE1's unicast peers if you have also configured the optional keyword **unicast**.

The **unicast** keyword is optional, yet significant, as it enables the PE devices to place unicast advertisements from the CE devices in the unicast BGP table of the PE devices. Therefore, route advertisements from CE devices populates both unicast and multicast BGP tables, else CE device's routes only populate the PE device's multicast BGP table.

Note You must also enable address-family under the compatible multicast address-family for VRF aware BGP translate-update feature to function as designed.

How To Configure VRF Aware BGP Translate-Update

Configuring VRF Aware BGP Translate-Update

Perform this task to configure VRF aware BGP translate-update feature:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4** [**mdt** | **tunnel** | {**multicast** | **unicast**} [**vrf** *vrf-name*] | **vrf** *vrf-name*]
5. **neighbor** *peer-group-name* **peer-group**
6. **neighbor** {*ipv4-addr* | *ipv6-addr* | *peer-group-name*} **remote-as** *autonomous-system-number*
7. **neighbor** {*ipv4-addr* | *ipv6-addr*} **peer-group** *peer-group-name*
8. **neighbor** {*ipv4-addr* | *ipv6-addr* | *peer-group-name*} **activate**
9. **neighbor** {*ipv4-address* | *ipv6-address*} **translate-update multicast** [**unicast**]
10. **end**
11. **show bgp vpnv4 multicast** {**all** | **vrf** *vrf-name* | **rd** *route-distinguisher*}
12. **show ip route multicast vrf** *vrf-name*
13. **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	**router bgp** *as-number* **Example:** `Device(config)# router bgp 65000`	Enters router configuration mode and creates a BGP routing process.
Step 4	**address-family ipv4** [**mdt** | **tunnel** | {**multicast** | **unicast**} [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config)# address-family ipv4 vrf v1`	Enters address family configuration mode to configure a routing session using standard IP Version 4 (IPv4) address prefixes.
Step 5	**neighbor** *peer-group-name* **peer-group** **Example:** `Device(config-af)# neighbor n2 peer-group`	Creates a BGP or multiprotocol BGP peer group.
Step 6	**neighbor** {*ipv4-addr* | *ipv6-addr* | *peer-group-name*} **remote-as** *autonomous-system-number* **Example:** `Device(config-af)# neighbor n2 remote-as 4`	Adds an entry to the BGP or multiprotocol BGP neighbor table.
Step 7	**neighbor** {*ipv4-addr* | *ipv6-addr*} **peer-group** *peer-group-name* **Example:** `Device(config-af)# neighbor 10.1.1.1 peer-group n2`	Configures a BGP neighbor to be a member of a peer group.
Step 8	**neighbor** {*ipv4-addr* | *ipv6-addr* | *peer-group-name*} **activate** **Example:** `Device(config-af)# neighbor 10.1.1.1 activate`	Enables exchange of information with a BGP neighbor.
Step 9	**neighbor** {*ipv4-address* | *ipv6-address*} **translate-update multicast** [**unicast**] **Example:** `Device(config-af)# neighbor 10.1.1.1 translate-update multicast unicast`	Enables multicast routing on devices, which are not capable of multicast BGP (mBGP) routing.
Step 10	**end** **Example:** `Device(config-af)# end`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 11	**show bgp vpnv4 multicast** {**all** | **vrf** *vrf-name* | **rd** *route-distinguisher*} **Example:** `Device# show bgp vpnv4 mul vrf v1 summary`	Displays Virtual Private Network Version 4 (VPNv4) multicast entries in a BGP table.
Step 12	**show ip route multicast vrf** *vrf-name* **Example:** `Device# show ip route multicast vrf v1`	Displays the IP routing table associated with a specific multicast VPN routing and forwarding (VRF) instance.
Step 13	**show running-config** **Example:** `Device# show running-config`	Displays the running configuration on the device.

Removing the VRF Aware BGP Translate-Update Configuration

Perform this task to disable the VRF aware BGP translate-update feature:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4** [**mdt** | **tunnel** | {**multicast** | **unicast**} [**vrf** *vrf-name*] | **vrf** *vrf-name*]
5. **no neighbor** {*ipv4-address* | *ipv6-address*} **translate-update multicast** [**unicast**]
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.
Step 3	**router bgp** *as-number* **Example:**	Enters router configuration mode and creates a BGP routing process.

	Command or Action	Purpose
	`Device(config)# router bgp 65000`	
Step 4	**address-family ipv4** [**mdt** | **tunnel** | {**multicast** | **unicast**} [**vrf** *vrf-name*] | **vrf** *vrf-name*] **Example:** `Device(config)# address-family ipv4 vrf v1`	Enters address family configuration mode to configure a routing session using standard IP Version 4 (IPv4) address prefixes.
Step 5	**no neighbor** {*ipv4-address* | *ipv6-address*} **translate-update multicast** [**unicast**] **Example:** `Device(config-af)# no neighbor 10.1.1.1 translate-update multicast unicast`	Disables multicast routing on devices, which are not capable of multicast BGP (mBGP) routing.
Step 6	**end** **Example:** `Device(config-af)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Displays the running configuration on the device.

Configuration Examples for VRF Aware BGP Translate-Update

Example: Configuring VRF aware BGP Translate-Update

The following example shows how to configure the translate-update feature for an IPv4 VRF address-family named v1 and BGP neighbor n2 peer-group for VRF configuration:

Note: Peer-template configuration for BGP neighbor is not supported for this feature due to conflicts with the earlier versions of Cisco software.

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# address-family ipv4 vrf v1
Device(config-router-af)# neighbor n2 peer-group
Device(config-router-af)# neighbor n2 remote-as 4
Device(config-router-af)# neighbor 10.1.1.1 peer-group n2
Device(config-router-af)# neighbor 10.1.1.1 activate
```

```
Device(config-router-af)# neighbor 10.1.1.1 translate-update multicast unicast
Device(config-router-af)# end
```

The following is sample output from the **show bgp vpnv4 multicast vrf** command. As the VRF aware BGP translate-update feature is configured, the state of the neighbor displays “NoNeg”:

```
Device# show bgp vpnv4 multicast vrf v1 summary

BGP router identifier 10.1.3.1, local AS number 65000
BGP table version is 8, main routing table version 8
7 network entries using 1792 bytes of memory
8 path entries using 960 bytes of memory
5/3 BGP path/bestpath attribute entries using 1280 bytes of memory
3 BGP AS-PATH entries using 88 bytes of memory
2 BGP extended community 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 4168 total bytes of memory
BGP activity 23/2 prefixes, 33/9 paths, scan interval 60 secs

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.1.1.1        4            4       5      10        1    0    0 00:01:10 (NoNeg)

10.1.3.2        4            2      12      10        8    0    0 00:01:33
```

The following is sample output from the **show ip route multicast vrf** command:

Note The routes configured using the translate-update feature does not have the “+” symbol against the prefixes in the Routing Information Base (RIB) table. Appearance of the symbol in the first entry indicates that the unicast route has leaked into the multicast table. However, the second entry is a translate-update route, which appears to be a multicast route.

```
Device# show ip route multicast vrf v1

B   +    10.1.1.0/24 [20/0] via 10.1.1.1 (v1), 00:00:08

B        10.1.1.0/24 [20/0] via 10.1.1.1 (v1), 00:00:42
```

The following is sample output from the **show running-config** command:

Note The provider-edge (PE) device must activate its BGP neighbor under the multicast address-family even though the neighbor is not capable of multicast routing. If the unicast address-family identifier has the route-map configured and multicast address-family identifier has no route-map configured, the unicast route-map controls the route under the unicast table but not the route under multicast table.

```
Device# show running-config

address-family ipv4 vrf v1
redistribute connected
redistribute static
neighbor 10.1.1.1 remote-as 4
neighbor 10.1.1.1 activate
```

```
neighbor 10.1.1.1 translate-update multicast unicast
neighbor 10.1.1.1 remote-as 4
neighbor 10.1.1.1 activate
exit-address-family
!
address-family ipv4 multicast vrf v1
redistribute connected
redistribute static
neighbor 10.1.1.1 remote-as 4
neighbor 10.1.1.1 activate
neighbor 10.1.1.1 soft-reconfiguration inbound
neighbor 10.1.1.1 route-map x in
exit-address-family
```

Note The “neighbor 10.1.1.1 soft-reconfiguration inbound” and the “ neighbor 10.1.1.1 route-map x in” field in the output indicate that only the routes in the BGP multicast table are affected.

The following is sample output from the **show running-config** command when you configure a neighbor under different address-families:

Note Configuring the BGP neighbor under different address-families manipulates the unicast routes and multicast routes advertised to the neighbor.

Configuration for IPv4/IPv6 unicast address-family:

```
Device# show running-config

address-family ipv4
neighbor 20.2.2.1 activate
neighbor 20.2.2.1 translate-update multicast unicast
exit-address-family
!
address-family ipv4 multicast
neighbor 20.2.2.1 activate
exit-address-family
!
```

Configuration for IPv4/IPv6 VRF unicast address-family:

```
Device# show running-config

address-family ipv4 vrf v1
neighbor 20.2.2.1 remote-as 4
neighbor 20.2.2.1 activate
neighbor 20.2.2.1 translate-update multicast unicast
exit-address-family
!
address-family ipv4 multicast vrf v1
neighbor 20.2.2.1 remote-as 4
neighbor 20.2.2.1 activate
exit-address-family
!
```

The following is sample configuration of the translate-update feature from a device with the old version of Cisco Software. The neighbor, in this case, is configured for IPv4/IPv6 unicast address-family, without running the **address-family** command:

Configuration in the old format, without an address-family configured:

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# neighbor 20.2.2.1 remote-as 4
Device(config-router)# neighbor 20.2.2.1 translate-update nlri ipv4 multicast unicast
Device(config-router-af)# end
```

Configuration in the new format, without an address-family configured:

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# neighbor 20.2.2.1 remote-as 4
Device(config-router)# neighbor 20.2.2.1 translate-update nlri multicast unicast
Device(config-router-af)# end
```

Example: Removing VRF aware BGP Translate-Update Configuration

The following example shows how to disable the VRF aware BGP translate-update feature for an IPv4 VRF address-family named v1 and BGP neighbor n2 peer-group for VRF:

Note Disabling the translate-update configuration for a neighbor deletes the pseudo multicast neighbor and flaps the session, similar to removing the neighbor from a multicast session:

```
Device> enable
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# address-family ipv4 vrf v1
Device(config-router-af)# no neighbor 10.1.1.1 translate-update multicast unicast
Device(config-router-af)# end
```

The following output displays the debug logs after you disable the translate-update feature on the neighbor:

```
*Nov 20 07:09:15.902: %BGP_SESSION-5-ADJCHANGE:
neighbor 2.2.2.1 IPv4 Multicast vpn vrf v1 topology base removed from session  Neighbor
deleted
*Nov 20 07:09:15.902: %BGP-5-ADJCHANGE:
neighbor 2.2.2.1 vpn vrf v1 Down Neighbor deleted
*Nov 20 07:09:15.902: %BGP_SESSION-5-ADJCHANGE:
neighbor 2.2.2.1 IPv4 Unicast vpn vrf v1 topology base removed from session  Neighbor deleted
*Nov 20 07:09:16.877: %BGP-5-ADJCHANGE:
neighbor 2.2.2.1 vpn vrf v1 Up
```

The following is sample output from the **show running-config** command:

Note The associated neighbor 10.1.1.1 is removed even from the nonvolatile generation (NVGEN) after translate-update is disabled on that neighbor.

```
Device# show running-config

address-family ipv4 vrf v1
redistribute connected
redistribute static
neighbor 10.1.1.1 remote-as 4
neighbor 10.1.1.1 activate
exit-address-family
!
address-family ipv4 multicast vrf v1
redistribute connected
redistribute static
exit-address-family
```

Additional References for VRF Aware BGP Translate-Update

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 website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for VRF Aware BGP Translate-Update

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 78: Feature Information for VRF Aware BGP Translate-Update

Feature Name	Releases	Feature Information
VRF aware BGP Translate-Update	15.4(2)T	The VRF aware BGP translate-update feature enables multicast forwarding on those customer-edge (CE) devices, which have an older version of Cisco software that does not support multicast BGP (mBGP) routing. The following command was introduced: **neighbor translate-update**

CHAPTER 66

BGP Accumulated IGP

The BGP Accumulated IGP feature is an optional nontransitive Border Gateway Protocol (BGP) path attribute. The attribute type code for the accumulated interior gateway protocol (AIGP) attribute is assigned by the Internet Assigned Numbers Authority (IANA). The value field of the AIGP attribute is defined as a set of type, length, value (TLV) elements. The AIGP TLV contains the AIGP metric.

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 Accumulated IGP

Overview of BGP Accumulated IGP

The BGP Accumulated IGP feature is required to simulate the current Open Shortest Path First (OSPF) behavior of computing the distance associated with a path. OSPF or Label Distribution Protocol (LDP) carries the prefix or label information only in the local area. Then, Border Gateway Protocol (BGP) carries the prefix or label to all the remote areas by redistributing the routes into BGP at area boundaries. The routes or labels are then advertised using label-switched paths (LSP). The next-hop for the route is changed at each Area Border Router (ABR) to a local device, which removes the need to leak OSPF routes across area boundaries. The bandwidth available on each of the core links is mapped to the OSPF cost; therefore, it is imperative that

BGP carries this cost correctly between each of the provider edge (PE) devices. This functionality is achieved by using the BGP Accumulated IGP feature.

You need to enable accumulated interior gateway protocol (AIGP) processing for internal Border Gateway Protocol (iBGP) and external Border Gateway Protocol (eBGP) neighbors to carry the AIGP attribute. Neighbors configured with the AIGP attribute are put in a separate update group from other iBGP neighbors. A separate update group is required for neighbors that are enabled to send the AIGP value to cost community. BGP needs to translate the AIGP attribute to the cost community or multi-exit discriminator (MED) and attach it to the route before advertising to legacy.

When BGP installs AIGP attribute routes into the routing information base (RIB), it adds the AIGP cost with the next-hop cost. If the next-hop is a nonrecursive IGP route, BGP sets the AIGP metric to the received AIGP value and the first hop IGP metric to the next-hop. If the next-hop is a recursive route with the AIGP metric, BGP adds the received AIGP metric to the next-hop AIGP metric.

Sending and Receiving BGP Accumulated IGP

When a session receives a prefix with the accumulated interior gateway protocol (AIGP) attribute and is not configured to receive AIGP information, the session discards the AIGP attribute and processes the remainder of the update message, and then it passes the AIGP attribute to other BGP peers. The route is then installed into the routing information base (RIB) and the prefix is sent with the AIGP attribute to all the AIGP-enabled neighbors. The AIGP attribute value is not updated if the next-hop of the route is not changed by the device before advertising it to the neighbor. If the device changes the next-hop of the route, it recalculates the AIGP attribute value by adding the next-hop metric to the received AIGP attribute value.

Originating Prefixes with Accumulated IGP

Origination of routes with the accumulated interior gateway protocol (AIGP) metric is controlled by configuration. AIGP attributes are attached to redistributed routes that satisfy the following conditions:

- The protocol redistributing the route is enabled for AIGP.
- The route is an interior gateway protocol (IGP) route redistributed into Border Gateway Protocol (BGP). The value assigned to the AIGP attribute is the value of the IGP next-hop to the route or as set by a route policy.
- The route is a static route redistributed into BGP. The value assigned is the value of the next-hop to the route or as set by a route policy.
- The route is imported into BGP through a network statement. The value assigned is the value of the next-hop to the route or as set by a route policy.
- The inbound or outbound route map also creates an AIGP attribute route map using the **set aigp-metric** command.

How to Configure BGP Accumulated IGP

Configuring AIGP Metric Value

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family ipv4** [**unicast** | **multicast** | **vrf** *vrf-name*]
5. **redistribute** *protocol* *autonomous-system-number* **route-map** *map-tag*
6. **network** *network-id* **route-map** *map-tag*
7. **exit**
8. **route-map rtmap**
9. **set aigp-metric** [**igp-metric** | *value*]
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** *as-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified 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.
Step 5	**redistribute** *protocol* *autonomous-system-number* **route-map** *map-tag* **Example:**	Redistributes routes from one routing domain to another routing domain.

	Command or Action	Purpose
	`Device(config-router-af)# redistribute bgp 100 route-map rtmap`	
Step 6	**network** *network-id* **route-map** *map-tag* **Example:** `Device(config-router-af)# network 10.1.1.1 route-map rtmap`	Specifies the networks to be advertised by the Border Gateway Protocol (BGP) routing process.
Step 7	**exit** **Example:** `Device(config-router-af)# exit`	Exits address family configuration mode and returns to global configuration mode.
Step 8	**route-map rtmap** **Example:** `Device(config)# route-map rtmap`	Enters route map configuration mode.
Step 9	**set aigp-metric [igp-metric |** *value***]** **Example:** `Device(config-route-map)# set aigp-metric igp-metric`	Specifies a metric value for the accumulated interior gateway protocol (AIGP) attribute. The manual metric value range is from 0 to 4294967295.
Step 10	**end** **Example:** `Device(config-route-map)# end`	Exits route map configuration mode and enters privileged EXEC mode.

Enabling Send and Receive for an AIGP Attribute

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family {ipv4 | ipv6} [unicast]**
5. **neighbor** *ip-address* **aigp**
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family {ipv4 | ipv6}** [**unicast**] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family and enters address family configuration mode.
Step 5	**neighbor** *ip-address* **aigp** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 aigp`	Enables send and receive of the AIGP attribute per neighbor.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuring BGP Accumulated IGP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *as-number*
4. **address-family {ipv4 | ipv6}** [**unicast**]
5. **neighbor** *ip-address* **aigp** **[send {cost-community** *community-id* **poi {igp-cost | pre-bestpath} [transitive]} | med]**
6. **end**

DETAILED STEPS

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

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *as-number* **Example:** `Device(config)# router bgp 40000`	Enters router configuration mode for the specified routing process.
Step 4	**address-family {ipv4 | ipv6}** [**unicast**] **Example:** `Device(config-router)# address-family ipv4 unicast`	Specifies the IPv4 or IPv6 address family and enters address family configuration mode.
Step 5	**neighbor** *ip-address* **aigp [send {cost-community** *community-id* **poi {igp-cost | pre-bestpath} [transitive]} | med]** **Example:** `Device(config-router-af)# neighbor 192.168.1.1 aigp send med`	Translates the AIGP attribute to MED and attaches it to the route before advertising to legacy provider edge (PE) devices.
Step 6	**end** **Example:** `Device(config-router-af)# end`	Exits address family configuration mode and returns to privileged EXEC mode.

Configuration Examples for BGP Accumulated IGP

Example: Configuring AIGP Metric Value

The following is a sample configuration for originating prefixes with the accumulated internal gateway protocol (AIGP) metric attribute:

```
Device# configure terminal
Device(config)# router bgp 40000
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# redistribute bgp 100 route-map rtmap
Device(config-router-af)# network 10.1.1.1 route-map rtmap
Device(config-router-af)# exit
Device(config)# route-map rtmap
```

```
Device(config-route-map)# set aigp-metric igp-metric
Device(config-route-map)# end
```

Example: Enabling Send and Receive for an AIGP Attribute

The following example shows how to enable AIGP send and receive capability in address family configuration mode:

```
Device# configure terminal
Device(config)# router bgp 40000
Device(config-router)# address-family ipv4 unicast
Device(config-router-af)# neighbor 192.168.1.1 aigp
Device(config-router-af)# exit
```

Example: Configuring BGP Accumulated IGP

In the following example, the device belongs to autonomous system 65000 and is configured to send the cost-community attribute to its neighbor at IP address 172.16.70.23:

```
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# address-family ipv4 multicast
Device(config-router-af)# neighbor 172.16.70.23 aigp send cost-community 100 poi igp-cost
transitive
Device(config-router-af)# exit
```

In the following example, the device belongs to autonomous system 65000 and is configured to send the MED attribute to its neighbor at IP address 172.16.70.23:

```
Device# configure terminal
Device(config)# router bgp 65000
Device(config-router)# address-family ipv4 multicast
Device(config-router-af)# neighbor 172.16.70.23 aigp send med
Device(config-router-af)# exit
```

Additional References for BGP Accumulated IGP

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 website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP Accumulated IGP

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 79: Feature Information for BGP Accumulated IGP

Feature Name	Releases	Feature Information
BGP Accumulated IGP	15.4(3)T	The BGP Accumulated IGP feature is an optional nontransitive Border Gateway Protocol (BGP) path attribute. The attribute type code for the accumulated interior gateway protocol (AIGP) attribute is assigned by the IANA. The value field of the AIGP attribute is defined as a set of type, length, value (TLV) elements. The AIGP TLV contains the AIGP metric. The following commands were introduced: **aigp**, **aigp send cost-community**, **aigp send med**, **bgp bestpath aigp ignore**, **set aigp-metric**

CHAPTER 67

BGP AS-Override Split-Horizon

The BGP AS-Override Split-Horizon feature enables a Provider Edge (PE) device using split-horizon to avoid advertisement of routes propagated by a Customer Edge (CE) device to the same CE device. The BGP AS-Override Split-Horizon feature also enables a PE or CE device to send route updates to a specific PE or CE device in the same replication group.

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 AS-Override Split-Horizon

BGP AS-Override Split-Horizon Overview

When you configure split-horizon on a device, the Provider Edge (PE) device may advertise routes propagated from a Customer Edge (CE) device to the same CE device. The BGP AS-Override Split Horizon feature groups all the BGP neighbors into separate replication-groups, even when they are in the same update-group, and ensures that the route updates propagated from a CE device are not sent to the same CE device.

The BGP AS-Override Split Horizon feature enables a PE or CE device to selectively send and block updates to one or more neighboring PE or CE devices in the same update-group. The PE or CE device sends or blocks a message to a neighboring PE or CE device based on the type of the message and on whether the originator of the message matches the router ID of the PE or CE device.

How to Configure BGP AS-Override Split-Horizon

Configuring BGP AS-Override Split-Horizon

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system-number*
4. **address family ipv4 vrf** *vrf-name*
5. **neighbor** *ip-address* **remote-as** *autonomous-system-number*
6. **neighbor** *ip-address* **activate**
7. **neighbor** *ip-address* **as-override split-horizon**
8. Repeat Step 5 to Step 7 to enable split-horizon for different neighbors in a virtual routing and forwarding (VRF) instance.
9. **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 21`	Configures the Border Gateway Protocol (BGP) routing process and enters router configuration mode.
Step 4	**address family ipv4 vrf** *vrf-name* **Example:** `Device(config-router)# address-family ipv4 vrf vrf1`	Specifies the name of the VPN routing and forwarding (VRF) instance to associate with subsequent IPv4 address family configuration mode commands and enters address-family configuration mode.

	Command or Action	Purpose
Step 5	**neighbor** *ip-address* **remote-as** *autonomous-system-number* **Example:** `Device(config-router-af)# neighbor 192.0.2.1 remote-as 1`	Configures peering with a BGP neighbor in the specified autonomous system.
Step 6	**neighbor** *ip-address* **activate** **Example:** `Device(config-router-af)# neighbor 192.0.2.1 activate`	Enables the neighbor to exchange prefixes for the IPv4 address family with the local device.
Step 7	**neighbor** *ip-address* **as-override split-horizon** **Example:** `Device(config-router-af)# neighbor 192.0.2.1 as-override split-horizon`	Enables split-horizon per neighbor in a VRF instance.
Step 8	Repeat Step 5 to Step 7 to enable split-horizon for different neighbors in a virtual routing and forwarding (VRF) instance.	—
Step 9	**end** **Example:** `Device(config-router-af)# end`	Exits router address-family configuration mode and enters privileged EXEC mode.

Verifying BGP AS-Override Split-Horizon

SUMMARY STEPS

1. **enable**
2. **show ip bgp vpn4 all update-group**
3. **show ip bgp vpnv4 all neighbors** *ip-address*
4. **show ip bgp vpnv4 all neighbors** *ip-address* **policy**

DETAILED STEPS

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

	Command or Action	Purpose
Step 2	**show ip bgp vpn4 all update-group** **Example:** `Device# show ip bgp vpn4 all update-group`	Displays information on update groups.
Step 3	**show ip bgp vpnv4 all neighbors** *ip-address* **Example:** `Device# show ip bgp vpnv4 all neighbors 192.0.2.1`	Displays details about neighbor connections.
Step 4	**show ip bgp vpnv4 all neighbors** *ip-address* **policy** **Example:** `Device# show ip bgp vpnv4 all neighbors 192.0.2.1 policy`	Displays neighbor policies per address-family.

Configuration Examples for BGP AS-Override Split-Horizon

Example: BGP AS-Override Split-Horizon Configuration

```
Device> enable
Device# configure terminal
Device(config)# router bgp 21
Device(config-router)# address-family ipv4 vrf vrf1
Device(config-router-af)# neighbor 192.0.2.1 remote-as 1
Device(config-router-af)# neighbor 192.0.2.1 activate
Device(config-router-af)# neighbor 192.0.2.1 as-override split-horizon
Device(config-router-af)# neighbor 198.51.100.1 remote-as 1
Device(config-router-af)# neighbor 198.51.100.1 activate
Device(config-router-af)# neighbor 198.51.100.1 as-override split-horizon
Device(config-router-af)# end
```

Example: Verifying BGP AS-Override Split-Horizon

Sample output for the show ip bgp vpn4 all update-group command

To display information about update groups, use the **show ip bgp vpn4 all update-group** command in privileged EXEC mode.

```
Device> enable
Device# show ip bgp vpn4 all update-group
BGP version 4 update-group 3, external, Address Family: VPNv4 Unicast
  BGP Update version : 5/0, messages 0 active RGs: 2 <<<<<<<<<<<<<<
  Overrides the neighbor AS 1 with my AS before sending updates
  Topology: blue, highest version: 5, tail marker: 5
  Format state: Current working (OK, last not in list)
                Refresh blocked (not in list, last not in list)
  Update messages formatted 1, replicated 2, current 0, refresh 0, limit 1000
  Number of NLRIs in the update sent: max 4, min 0
```

```
  Minimum time between advertisement runs is 0 seconds
  Has 2 members:
   192.0.2.1         198.51.100.1
```

Sample output for the show ip bgp vpnv4 all neighbors *ip-address* command

To display details about neighbor connections, use the **show ip bgp vpnv4 all neighbors** *ip-address* command in privileged EXEC mode.

```
Device> enable
Device# show ip bgp vpnv4 all neighbors 209.165.200.228
BGP neighbor is 209.165.200.228,  vrf vrf1,  remote AS 1, external link
  BGP version 4, remote router ID 209.165.201.28
  BGP state = Established, up for 00:01:26
  Last read 00:00:35, last write 00:00:28, hold time is 180, keepalive interval is 60 seconds

  Neighbor sessions:
    1 active, is not multisession capable (disabled)
  Neighbor capabilities:
    Route refresh: advertised and received(new)
    Four-octets ASN Capability: advertised and received
    Address family IPv4 Unicast: advertised and received
    Enhanced Refresh Capability: advertised and received
    Multisession Capability:
    Stateful switchover support enabled: NO for session 1
  Message statistics:
    InQ depth is 0
    OutQ depth is 0

                         Sent       Rcvd
    Opens:                  1          1
    Notifications:          0          0
    Updates:                6          2
    Keepalives:             3          3
    Route Refresh:          0          0
    Total:                 12          6
  Default minimum time between advertisement runs is 0 seconds

 For address family: VPNv4 Unicast
  Translates address family IPv4 Unicast for VRF vrf1
  Session: 209.165.200.228
  BGP table version 40, neighbor version 40/0
  Output queue size : 0
  Index 1, Advertise bit 1
  1 update-group member
  Overrides the neighbor AS with my AS before sending updates
  Split horizon processing before sending updates
  Slow-peer detection is disabled
  Slow-peer split-update-group dynamic is disabled
                                 Sent       Rcvd
  Prefix activity:               ----       ----
    Prefixes Current:              10          2 (Consumes 160 bytes)
    Prefixes Total:                10          2
    Implicit Withdraw:              0          0
    Explicit Withdraw:              0          0
    Used as bestpath:             n/a          2
    Used as multipath:            n/a          0
Outbound    Inbound
  Local Policy Denied Prefixes:    --------    -------
    Total:                                0          0
  Number of NLRIs in the update sent: max 5, min 0
  Last detected as dynamic slow peer: never
  Dynamic slow peer recovered: never
  Refresh Epoch: 1
```

```
  Last Sent Refresh Start-of-rib: 00:01:26
  Last Sent Refresh End-of-rib: 00:01:26
  Refresh-Out took 0 seconds
  Last Received Refresh Start-of-rib: never
  Last Received Refresh End-of-rib: never
                                    Sent       Rcvd
        Refresh activity:           ----       ----
          Refresh Start-of-RIB        1          0
          Refresh End-of-RIB          1          0

  Address tracking is enabled, the RIB does have a route to 209.165.200.228
  Connections established 3; dropped 2
  Last reset 00:01:35, due to split-horizon config change of session 1
  Transport(tcp) path-mtu-discovery is enabled
  Graceful-Restart is disabled
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled
Mininum incoming TTL 0, Outgoing TTL 1
Local host: 209.165.200.225, Local port: 22789
Foreign host: 209.165.200.228, Foreign port: 179
Connection tableid (VRF): 2
```

Sample output for the show ip bgp vpnv4 all neighbors *ip-address* policy command

To display neighbor policies per address-family, use the **show ip bgp vpnv4 all neighbors** *ip-address* **policy** command in privileged EXEC mode.

```
Device> enable
Device# show ip bgp vpnv4 all neighbors 209.165.200.228
Neighbor: 209.165.200.228, Address-Family: VPNv4 Unicast (vrf1)
 Locally configured policies:
  as-override split-horizon
```

Additional References for BGP AS-Override Split-Horizon

Related Documents

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

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for BGP AS-Override Split-Horizon

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 80: Feature Information for BGP AS-Override Split-Horizon

Feature Name	Releases	Feature Information
BGP AS-Override Split-Horizon	15.5(2)T	The BGP AS-Override Split-Horizon feature enables a Provider Edge (PE) device using split-horizon to avoid advertisement of routes propagated by a Customer Edge (CE) device to the same CE device. The BGP AS-Override Split-Horizon feature also enables a PE or CE device to send route updates to specific PE or CE device in the same replication group. The following command was introduced or modified: **neighbor** *ip-address* **as-override split-horizon**.