

BGP Configuration Guide

First Published: 2026-02-20

Last Modified: 2026-07-17

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

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

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

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

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

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

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

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

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

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2026 Cisco Systems, Inc. All rights reserved.



Read Me First

Only supported features are documented. To confirm or clarify all the supported features for a platform, go to Cisco Feature Navigator.

Read Me First



CONTENTS

PREFACE

Read Me First iii

CHAPTER 1

Understanding BGP 1

- Feature History for BGP 1
- Understanding BGP 2
  - Benefits BGP 2
  - Restrictions BGP 3
  - Network Topology 3
- BGP Components 4
- Key Concepts 5
  - BGP Peers (Neighbours) 5
  - Autonomous Systems 6
  - Full Mesh Requirement in BGP 7
  - AS Path Attribute in BGP 7
  - BGP Route Attributes 7
  - BGP Route Exchange 8
  - Next-Hop Reachability and Synchronization in BGP 9
  - Support for CIDR and Route Aggregation in BGP 9
  - c-Nonstop Forwarding Awareness 10
  - BGP Policy Control 10
- How to configure BGP 10
- Configuring a BGP routing process 14
- Managing Routing Policy Changes 16
- Configuring BGP Decision Attributes 17
- Configuring BGP Filtering with Route Maps 20
- Configuring BGP Filtering by Access Lists and Neighbors 21

Configuring Prefix Lists for BGP Filtering 22
Configuring BGP Community Filtering 23
Configuring BGP Neighbors and Peer Groups 25
Configuring Aggregate Addresses in a Routing Table 29
Configuring Routing Domain Confederations 30
Configuring BGP Route Reflectors 32
Configuring Route Dampening 33
Conditionally Injecting BGP Routes 35
Configuring Peer Session Templates 37
Configuring a basic peer session template 38
Configuring Peer Session Template Inheritance with the inherit peer-session Command 39
Configuring Peer Session Template Inheritance with the neighbor inherit peer-session Command 41
Configuring Peer Policy Templates 42
Configuring Basic Peer Policy Templates 42
Configuring Peer Policy Template Inheritance with the inherit peer-policy Command 44
Configuring Peer Policy Template Inheritance with the neighbor inherit peer-policy Command 46
Configuring BGP Route Map Next-hop Self 48
Configuration Examples for BGP 51
Configuring Conditional BGP Route Injection 51
Configuring Peer Session Templates 51
Configuring Peer Policy Templates 52
Configuring BGP Route Map next-hop self 53
Monitoring and Maintaining BGP 53

CHAPTER 2 **Configure BGP Graceful Shutdown 55**
Feature history for BGP graceful shutdown 55
Information about BGP graceful shutdown 55
BGP graceful shutdown benefits 56
GSHUT community 56
Configure BGP GSHUT enhancement 56
Configure BGP graceful shutdown 58
Shut down a BGP link gracefully 58
Filter BGP routes based on the GSHUT community 60

Configure BGP GSHUT enhancement 62
Configuration examples for BGP graceful shutdown 64
Example: shut down a BGP link gracefully 64
Filter BGP routes based on the GSHUT community 64
Configure BGP GSHUT enhancement 65
Additional references 67

CHAPTER 3 **Configure BGP Large Community 69**
Feature history for BGP large community 69
Restrictions for the BGP large community 70
BGP large community feature 70
Large community lists 70
BGP large community attribute 71
Configure BGP large community 71
Enable BGP large community 71
Configure route-map with large community lists and matching a large community 73
Define BGP large community list 74
Configure the route-map to set BGP large communities 76
Delete large communities 77
Verify the configuration of the BGP large community 78
Troubleshoot large communities 79
Configuration example for BGP large community 79

CHAPTER 4 **Configuring BGP Support for 4-byte ASN 81**
Feature history for BGP support for 4-byte ASN 81
Information about BGP support for 4-byte ASN 81
BGP autonomous system number formats 82
Cisco implementation of 4-byte autonomous system numbers 84
Configure BGP support for 4-byte ASN 85
Configure a BGP routing process and peers using 4-byte autonomous system numbers 85
Modify the default output and regular expression match format for 4-byte autonomous system numbers 88
Configuration examples for BGP support for 4-byte ASN 91
Examples: configure a BGP routing process and peers using 4-byte autonomous system numbers 91

Examples: configure a VRF and set an extended community using a BGP 4-Byte autonomous system number 94

Additional references for BGP support for 4-byte ASN 95

CHAPTER 5 **Configuring BGP Monitoring Protocol** 97

Feature history for BGP Monitoring Protocol 97

BGP Monitoring Protocol 98

Prerequisites for BGP monitoring protocol 99

Configure a BGP Monitoring Protocol session 99

Configuration examples for BGP Monitoring Protocol 100

Configure BGP Monitoring Protocol servers 104

Configure BGP Monitoring Protocol on VRF neighbors 107

Verify BGP Monitoring Protocol 108

Monitor BGP Monitoring Protocol 109

Additional references for BGP Monitoring Protocol 109

CHAPTER 6 **Configuring BGP Next Hop Unchanged** 111

Feature history for BGP next hop unchanged 111

Information about BGP next hop unchanged 111

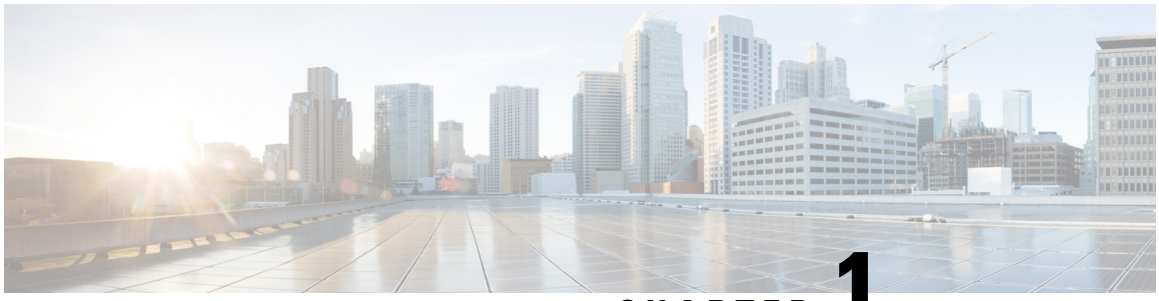
Restrictions for BGP next hop unchanged 112

How to configure BGP next hop unchanged 112

Configure the BGP next hop unchanged for an eBGP peer 112

Configure BGP next hop unchanged using route maps 114

Configuration example for BGP next hop unchanged for an eBGP peer 115



CHAPTER 1

Understanding BGP

- Feature History for BGP, on page 1
- Understanding BGP, on page 2
- BGP Components, on page 4
- Key Concepts, on page 5
- How to configure BGP, on page 10
- Configuring a BGP routing process, on page 14
- Managing Routing Policy Changes, on page 16
- Configuring BGP Decision Attributes, on page 17
- Configuring BGP Filtering with Route Maps, on page 20
- Configuring BGP Filtering by Access Lists and Neighbors, on page 21
- Configuring Prefix Lists for BGP Filtering, on page 22
- Configuring BGP Community Filtering, on page 23
- Configuring BGP Neighbors and Peer Groups, on page 25
- Configuring Aggregate Addresses in a Routing Table, on page 29
- Configuring Routing Domain Confederations, on page 30
- Configuring BGP Route Reflectors, on page 32
- Configuring Route Dampening, on page 33
- Conditionally Injecting BGP Routes, on page 35
- Configuring Peer Session Templates, on page 37

Feature History for BGP

This table provides release and platform support information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Table 1: Feature history for BGP

Release	Feature name and description	Supported platform
Cisco IOS XE 26.2.1ea	Border Gateway Protocol: The BGP feature support has been introduced.	Cisco C9550 Series Smart Switches

Release	Feature name and description	Supported platform
Cisco IOS XE 17.18.1	Border Gateway Protocol: The Border Gateway Protocol (BGP) is an exterior gateway protocol used to set up an inter-domain routing system that guarantees the loop-free exchange of routing information between autonomous systems.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

Understanding BGP

BGP stands for Border Gateway Protocol.

Here are some key characteristics of BGP:

- BGP is an Exterior Gateway Protocol (EGP).
- Its purpose is to establish an interdomain routing system.
- It ensures the loop-free exchange of routing information between autonomous systems (AS).
- AS is a collection of routers under a single administration, typically running Interior Gateway Protocols (IGPs) like RIP or OSPF internally and it correctly cites the defining RFCs (1163, 1267, and 1771).
- Path-Vector Protocol: Unlike interior gateway protocols (IGPs) like OSPF or EIGRP that use metrics like bandwidth or delay, BGP makes routing decisions based on paths, attributes, and policies. It carries a list of AS numbers that a route has traversed.
- Inter-Domain Routing: BGP's primary role is to route traffic between different autonomous systems.
- Policy-Based Routing: BGP allows network administrators to implement complex routing policies, such as traffic engineering, peering agreements, and security filtering.
- Reliable Transport: BGP uses TCP (port 179) as its transport protocol, ensuring reliable delivery of routing updates

Benefits BGP

The benefits of BGP (Border Gateway Protocol) include:

- Scalability for Inter-AS Routing: BGP enables routing between different autonomous systems (ASes), supporting large-scale internet routing by exchanging reachability information across AS boundaries.
- Loop Prevention: BGP uses the AS path attribute to prevent routing loops by tracking the sequence of ASes a route has traversed.
- Policy Control: BGP allows fine-grained control over routing decisions and path selection through various attributes, enabling enforcement of routing policies at the AS level.
- Support for Both IBGP and EBGP: BGP supports internal routing within an AS (IBGP) and external routing between ASes (EBGP), providing flexibility in network design.

- Route Aggregation and CIDR Support: BGP Version 4 supports Classless Inter-Domain Routing (CIDR), allowing route aggregation to reduce routing table size and improve efficiency.
- Reliable Transport: BGP uses TCP (port 179) for reliable session establishment and maintenance between peers.
- Incremental Updates: After the initial full routing table exchange, BGP sends only incremental updates, reducing bandwidth and processing overhead.
- Nonstop Forwarding Awareness: When combined with features like Graceful Restart, BGP can maintain forwarding during route processor failover or software upgrades, enhancing network stability.

Restrictions BGP

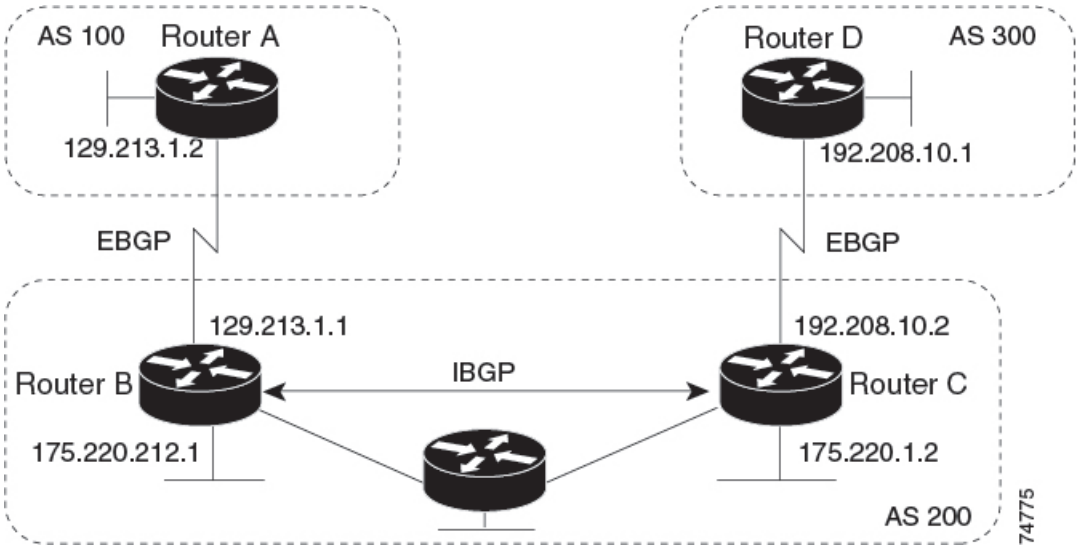
- **BGP Graceful Restart and xFSU**: When configuring **bgp graceful-restart** on xFSU-capable devices, the BGP hold time must exceed the device-specific minimum graceful restart time. If the hold time is lower or an unsupported hold timer is received from a peer, BGP will not be supported during an xFSU event, even if the value is accepted.
- **Layer 3 Forwarding Delay**: Upon device startup or after executing the **clear ip bgp** command, Layer 3 forwarding will be delayed for approximately 80 seconds until routing tables are fully populated. You can verify the routing table population status using the **show ip bgp ip-address** command.
- The routing tables require around 80 seconds for population. You can use the show ip bgp ip-address command, in privileged EXEC mode, to check whether the routing tables are populated or not.
-
-

Network Topology

Routers within the same autonomous system (AS) that exchange BGP updates run internal BGP (IBGP). Routers in different autonomous systems that exchange BGP updates run external BGP (EBGP). Most configuration commands are the same for EBGP and IBGP, but EBGP exchanges updates between autonomous systems, while IBGP exchanges updates within an AS. Figure 1 illustrates a network running both EBGP and IBGP.

Before exchanging information with an external AS, BGP ensures that networks within the AS can be reached by defining internal BGP peering among routers within the AS and by redistributing BGP routing information to IGPs that run within the AS, such as IGRP and OSPF.

Routers running a BGP routing process are called BGP speakers. BGP uses Transmission Control Protocol (TCP) on port 179 as its transport protocol. Two BGP speakers with a TCP connection for exchanging routing information are known as peers or neighbors. In Figure 1, Routers A and B are BGP peers. Routers B and C, and Routers C and D, are also peers. he routing information consists of a series of AS numbers that specify the full path to the destination network. BGP uses this information to build a loop-free map of autonomous systems.



The network has these characteristics:

- Routers A and B run EBGP, while Routers B and C run IBGP. EBGP peers are directly connected, but IBGP peers do not need to be. If an IGP is running that allows the neighbors to reach each other, IBGP peers can be indirectly connected.
- All BGP speakers within an AS must establish peer relationships with each other, forming a full logical mesh. BGP4 provides two techniques to reduce this requirement: confederations and route reflectors.
- AS 200 is a transit AS for AS 100 and AS 300; it transfers packets between AS 100 and AS 300.

BGP peers initially exchange their full routing tables, then send only incremental updates. They also exchange keepalive messages to ensure the connection is up and notification messages if there are errors or special conditions.

In BGP, each route consists of a network number, an autonomous system path (the list of autonomous systems the information has passed through), and other path attributes. The main function of BGP is to exchange network reachability information, including AS paths, with other BGP systems. This information helps determine AS connectivity, prunes routing loops, and enforces AS-level policy decisions.

A router or device running Cisco IOS will not select or use an IBGP route unless it has a route to the next-hop router and has received IGP synchronization, unless synchronization is disabled. When multiple routes are available, BGP bases its path selection on attribute values. SFor details about BGP attributes, refer to the “Configuring BGP Decision Attributes” section.

BGP Version 4 supports classless interdomain routing (CIDR), which reduces the size of routing tables by allowing the creation of aggregate routes and supernets. CIDR removes the concept of network classes in BGP and supports advertising IP prefixes.

BGP Components

- BGP Speaker (Router): A device running the BGP routing process that exchanges routing information with other BGP speakers.

- BGP Peers (Neighbors): Two BGP speakers that establish a TCP connection (port 179) to exchange routing updates. Peers can be:
  1. External BGP (EBGP): Peers in different autonomous systems (AS).
  2. Internal BGP (IBGP): Peers within the same AS.
- Autonomous System (AS): A collection of IP networks under a single administrative domain. BGP manages routing between and within ASes.
- BGP Routing Table: Stores the routes learned from BGP peers, including network prefixes and path attributes.
- Path Attributes: Information associated with each route that influences path selection, such as AS path, next-hop, local preference, and others.
- AS Path: A list of AS numbers that a route has traversed, used to prevent routing loops and assist in path selection.
- TCP Connection: BGP uses TCP (port 179) as its transport protocol to ensure reliable delivery of routing information.
- Route Reflectors and Confederations: Mechanisms to reduce the full mesh IBGP requirement by logically grouping BGP speakers.
- Next-Hop Reachability and Synchronization: Ensures that a router only uses IBGP routes if it can reach the next-hop and synchronization with the IGP is achieved (unless disabled).

Key Concepts

- BGP Peers (Neighbors)
- Autonomous Systems (AS)
- Full Mesh Requirement
- AS Path Attribute
- Route Attributes
- Route Exchange
- Next-Hop Reachability and Synchronization
- Support for Classless Inter-Domain Routing (CIDR) and Route Aggregation
- Nonstop Forwarding Awareness
- Policy Control

BGP Peers (Neighbours)

BGP peers, also known as neighbors, are routers that have established a TCP connection (on port 179) to exchange BGP routing information. These peers can be classified into two types:

- External BGP (eBGP) Peers: Routers that belong to different autonomous systems (AS) and exchange routing information across AS boundaries. Typically, eBGP peers are directly connected.
- Internal BGP (iBGP) Peers: Routers within the same AS that exchange BGP updates. iBGP peers do not need to be directly connected as long as an Interior Gateway Protocol (IGP) provides reachability between them.

Key characteristics of BGP peers include:

- Full Mesh Requirement: All iBGP peers within an AS must be logically fully meshed, meaning each iBGP router must establish a peering session with every other iBGP router. This requirement can be relaxed using route reflectors or confederations.
- BGP Speakers: Routers running BGP are called BGP speakers. They exchange full routing tables initially and then send incremental updates.
- Route Exchange: Peers exchange network reachability information, including AS path attributes, to build a loop-free interdomain routing topology.
- Peer Relationships: Peers maintain the connection by exchanging keepalive messages and use notification messages to signal errors or special conditions.
- Policy Application: BGP peer policies control route advertisement and acceptance, applied individually or via peer groups to simplify management.

Autonomous Systems

An Autonomous System (AS) is a collection of IP networks and routers under the control of a single organization that presents a common routing policy to the Internet. Each AS is assigned a unique AS number (ASN) which is used to identify it globally.

Key points about Autonomous Systems in BGP:

- Purpose: ASes enable the organization of the Internet into manageable routing domains. They allow routing policies to be applied consistently within the AS and control how routing information is exchanged with other ASes.
- Types of BGP Peering:
  - External BGP (eBGP): Peering between routers in different ASes. eBGP peers exchange routing information across AS boundaries.
  - Internal BGP (iBGP): Peering between routers within the same AS. iBGP peers exchange routing information internally to ensure consistent routing within the AS.
- AS Path Attribute: BGP uses the AS path attribute to record the sequence of ASes that routing information has traversed. This helps in loop prevention and policy enforcement.
- Transit AS: An AS that carries traffic between other ASes without being the source or destination. For example, AS 200 can be a transit AS between AS 100 and AS 300.
- Routing Policy: Each AS can implement its own routing policies, controlling how routes are advertised and accepted from other ASes.

Full Mesh Requirement in BGP

The Full Mesh Requirement in BGP states that all BGP speakers within the same Autonomous System (AS) must establish a peer relationship with each other. This means every iBGP router must be directly connected logically to every other iBGP router in the AS to exchange routing information.

Key points about the Full Mesh Requirement:

- Purpose: Ensures that routing information learned from an external BGP (eBGP) peer is propagated to all other routers within the AS.
- Logical Full Mesh: Even if routers are not physically directly connected, they must have a logical peer connection.
- Scalability Challenge: As the number of routers increases, the number of peer connections grows exponentially, making full mesh difficult to manage.
- Solutions to Reduce Full Mesh:
  - Route Reflectors: Allow some routers to reflect routes to others, reducing the need for a full mesh.
  - Confederations: Divide a large AS into smaller sub-ASes to reduce the number of iBGP peers.

AS Path Attribute in BGP

The AS Path Attribute is a fundamental path attribute in BGP that lists the sequence of Autonomous Systems (ASes) a route has traversed to reach a destination network. It is used primarily for:

- Loop Prevention: BGP uses the AS path to detect and prevent routing loops by checking if its own AS number appears in the AS path of a received route. If it does, the route is discarded.
- Path Selection: The AS path length (number of AS hops) is a key factor in BGP's route selection process. Routes with shorter AS paths are generally preferred.
- Policy Enforcement: Network administrators can apply routing policies based on AS path information to control route advertisement and acceptance.

The AS path attribute is carried in every BGP update message and is updated by each AS that propagates the route. When a BGP speaker advertises a route to an external peer, it prepends its own AS number to the AS path.

BGP Route Attributes

BGP uses several route attributes to determine the best path to a destination and to enforce routing policies. These attributes are carried in BGP update messages and influence route selection and advertisement. The main BGP route attributes include:

- Well-Known Mandatory Attributes (must be recognized and included in all BGP updates)
  - AS Path: Lists the sequence of Autonomous Systems a route has traversed; used for loop prevention and path selection.
  - Next Hop: Specifies the IP address of the next-hop router to reach the destination.
  - Origin: Indicates the origin of the route (IGP, EGP, or incomplete).

- Local Preference: Used within an AS to prefer an exit point for outbound traffic.
- Atomic Aggregate: Indicates that the route is an aggregate of multiple routes.
- Aggregator: Identifies the router that performed route aggregation

- Well-Known Discretionary Attributes (recognized by all BGP implementations but optional in updates)
  - Local Preference (also considered discretionary in some contexts).
- Optional Transitive Attributes (may be passed along to other BGP peers even if not recognized)
  - Community: Used to group routes and apply routing policies
  - Extended Community: Provides additional tagging capabilities.
- Optional Non-Transitive Attributes (used only between directly connected BGP peers)

Multi-Exit Discriminator (MED): Suggests preferred entry points into an AS from neighboring ASes.

BGP Route Exchange

Route Exchange in BGP refers to the process by which BGP peers (neighbors) share routing information to build and maintain a consistent and loop-free interdomain routing table. The key points about BGP route exchange are:

Summary

- Initial Full Table Exchange: When two BGP peers establish a session, they first exchange their entire BGP routing tables. This ensures that both peers have a complete view of the reachable networks known to each other.
- Incremental Updates: After the initial exchange, BGP peers send only incremental updates. These updates include new routes, withdrawn routes, or changes to existing routes, minimizing the amount of data transmitted and improving efficiency.
- TCP Transport: BGP uses TCP port 179 to establish a reliable connection between peers, ensuring ordered and error-checked delivery of routing updates.
- Keepalive Messages: To maintain the session, BGP peers periodically send keepalive messages. These confirm that the connection is still active and prevent session timeout.
- Notification Messages: If errors or special conditions occur, BGP peers send notification messages to inform each other and potentially terminate the session if necessary.
- Next-Hop Reachability: For a BGP route to be used, the router must have a route to the next-hop IP address, typically learned via an IGP or static route, ensuring proper forwarding.
- Synchronization with IGP: Unless synchronization is disabled, BGP requires that routes learned via IBGP are also present in the IGP before being used, to prevent routing loops.

This route exchange mechanism allows BGP to efficiently propagate routing information across autonomous systems while supporting policy control and loop prevention.

Workflow

1. •

When...	And...	Then...	And...

Result

What's next

Next-Hop Reachability and Synchronization in BGP

- Next-Hop Reachability: For a BGP route to be considered valid and used for forwarding, the router must have a route to the next-hop IP address specified in the BGP update. This next-hop is typically learned through an Interior Gateway Protocol (IGP) or a static route. Without reachability to the next-hop, the BGP route will not be installed in the routing table, ensuring proper packet forwarding and avoiding blackholing of traffic.
- Synchronization: This is a mechanism to prevent routing loops between BGP and the IGP. When synchronization is enabled, a BGP router will not advertise a route learned via Internal BGP (IBGP) to external peers unless that route is also present in the IGP. This ensures that all routers within the AS have consistent routing information before routes are advertised outside the AS. However, in many modern networks, synchronization is disabled because the IGP and BGP are managed separately, and route reflectors or confederations are used to scale IBGP.

Together, these mechanisms ensure that BGP routes are valid and consistent within and across autonomous systems, maintaining stable and loop-free routing environments.

Support for CIDR and Route Aggregation in BGP

- CIDR (Classless Inter-Domain Routing) Support: BGP fully supports CIDR, allowing the advertisement of IP prefixes of variable length rather than being restricted to classful boundaries. This capability enables more efficient use of IP address space and reduces the size of routing tables by allowing aggregation of routes.
- Route Aggregation: BGP supports route aggregation (also known as route summarization), which allows multiple contiguous IP prefixes to be combined into a single, summarized route advertisement. This reduces the number of routes exchanged between BGP peers, improving routing efficiency and scalability.
- How It Works:
  - Aggregation is configured on the BGP router to advertise a summarized route that represents multiple more specific routes.
  - The aggregated route can include attributes such as the AS path and next-hop, which are used to maintain routing consistency.
  - BGP also supports the use of the aggregate-address command to configure route aggregation and control attributes like the summary-only option to suppress more specific routes.
- Benefits

- Reduces routing table size.
- Limits the propagation of detailed routing information.
- Enhances network stability and performance.

c-Nonstop Forwarding Awareness

-
-
-

BGP Policy Control

Policy Control in BGP is a critical mechanism that allows network administrators to influence and control the routing decisions and the advertisement of routes between BGP peers. It provides flexibility to enforce routing policies based on various criteria.

Key aspects of BGP Policy Control include:

- Route Filtering: Control which routes are accepted from or advertised to BGP neighbors using prefix lists, route maps, or access control lists (ACLs). This helps in restricting unwanted routes and controlling routing updates.
- Route Maps: These are powerful tools used to match and modify route attributes such as AS path, next-hop, metric, local preference, and community values. Route maps enable conditional policy application.
- AS Path Filtering and Manipulation: Policies can filter routes based on AS path patterns or modify the AS path attribute to influence route selection and prevent routing loops.
- Local Preference: This attribute is used within an AS to prefer certain routes over others. Policy control can set local preference values to influence outbound traffic paths.
- Community Attributes: BGP communities are tags attached to routes that can be used to apply routing policies across multiple routers consistently.
- Next-Hop Control: Policies can modify the next-hop attribute to ensure reachability or influence routing decisions.
- Prefix Lists and Route Maps Combination: Often used together to create granular policies for route advertisement and acceptance.
- Policy Application Points: Policies can be applied inbound (on routes received from peers) or outbound (on routes advertised to peers).

These policy control mechanisms enable fine-grained management of BGP routing behavior, ensuring optimal routing, security, and compliance with organizational requirements.

How to configure BGP

This section provides configurational information about BGP.

Default Setting	
Aggregate address	Disabled: None defined.
AS path access list	None defined.
Auto summary	Disabled.
Best path	The router considers as-path in choosing a route and does not compare similar routes from external BGP peers. Compare router ID: Disabled.
BGP community list	Number: None defined. When you permit a value for the community number, the list defaults to an implicit deny for everything else that has not been permitted. Format: Cisco default format (32-bit number).
BGP confederation identifier/peers	Identifier: None configured. Peers: None identified.
BGP Fast external fallover	Enabled.
BGP local preference	100. The range is 0 to 4294967295 with the higher value preferred.

Default Setting	
BGP network	None specified; no backdoor route advertised.
BGP route dampening	Disabled by default. When enabled: Half-life is 15 minutes. Re-use is 750 (10-second increments). Suppress is 2000 (10-second increments). Max-suppress-time is 4 times half-life; 60 minutes.
BGP router ID	The IP address of a loopback interface if one is configured or the highest IP address configured for a physical interface on the router.
Default information originate (protocol or network redistribution)	Disabled.
Default metric	Built-in, automatic metric translations.

Default Setting	
Distance	External route administrative distance: 20 (acceptable values are from 1 to 255). Internal route administrative distance: 200 (acceptable values are from 1 to 255). Local route administrative distance: 200 (acceptable values are from 1 to 255).
Distribute list	In (filter networks received in updates): Disabled. Out (suppress networks from being advertised in updates): Disabled.
Internal route redistribution	Disabled.
IP prefix list	None defined.
Multi exit discriminator (MED)	Always compare: Disabled. Does not compare MEDs for paths from neighbors in different autonomous systems. Best path compare: Disabled. MED missing as worst path: Disabled.

Default Setting	
	Deterministic MED comparison is disabled.
Neighbor	Advertisement interval: 30 seconds for external peers; 5 seconds for internal peers.
	Change logging: Enabled.
	Conditional advertisement: Disabled.
	Default originate: No default route is sent to the neighbor.
	Description: None.
	Distribute list: None defined.
	External BGP multihop: Only directly connected neighbors are allowed.
	Filter list: None used.
	Maximum number of prefixes received: No limit.
	Next hop (router as next hop for BGP neighbor): Disabled.
	Password: Disabled.

Default Setting	
	Peer group: None defined; no members assigned.
	Prefix list: None specified.
	Remote AS (add entry to neighbor BGP table): No peers defined.
	Private AS number removal: Disabled.
	Route maps: None applied to a peer.
	Send community attributes: None sent to neighbors.
	Shutdown or soft reconfiguration: Not enabled.
	Timers: keepalive: 60 seconds; holdtime: 180 seconds.
	Update source: Best local address.
	Version: BGP Version 4.
	Weight: Routes learned through BGP peer: 0; routes sourced by the local router: 32768.
Default Setting	
NSF[1] Awareness	Disabled[2]. If enabled, allows Layer 3 switches to continue forwarding packets from a neighboring NSFcapable router during hardware or software changes.
Route reflector	None configured.
Synchronization (BGP and IGP)	Disabled.
Table map update	Disabled.
Timers	Keepalive: 60 seconds; holdtime: 180 seconds.

Default Setting	
NSF[1] Awareness	Disabled[2]. If enabled, allows Layer 3 switches to continue forwarding packets from a neighboring NSFcapable router during hardware or software changes.
Route reflector	None configured.
Synchronization (BGP and IGP)	Disabled.

Default Setting	
Table map update	Disabled.
Timers	Keepalive: 60 seconds; holdtime: 180 seconds.

[1] Nonstop Forwarding

[2] NSF Awareness can be enabled for IPv4 on switches with the Network Advantage license by enabling Graceful Restart.

Procedure

Use the task context details in this topic.

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.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp autonomous-system-number**

Example:

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

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

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

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

Example:

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

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

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

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

Example:

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

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

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

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

Step 6 **timers bgp keepalive holdtime**

Example:

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

(Optional) Sets BGP network timers.

Use the keepalive argument to specify the frequency, in seconds, with which the software sends keepalive messages to its BGP peer. By default, the keepalive timer is set to 60 seconds.

Use the holdtime argument to specify the interval, in seconds, after which the software, having not received a keepalive message, declares a BGP peer dead. By default, the holdtime timer is set to 180 seconds.

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

Example:

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

(Optional) Enables the automatic resetting of BGP sessions.

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

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

Example:

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

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

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

Step 9 **end**

Example:

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

Exits router configuration mode and enters privileged EXEC mode.

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

Example:

```
Device# show ip bgp
```

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

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

Managing Routing Policy Changes

To learn if a BGP peer supports the route refresh capability and to reset the BGP session:

Procedure

Step 1 **show ip bgp neighbors**

Example:

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

Displays whether a neighbor supports the route refresh capability. When supported, this message appears for the router: Received route refresh capability from peer.

Step 2 **clear ip bgp**{*|*address*|*peer-group-name*}

Example:

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

Resets the routing table on the specified connection.

- Enter an asterisk (*) to specify that all connections be reset.
- Enter an IP address to specify the connection to be reset.
- Enter a peer group name to reset the peer group.

Step 3 **clear ip bgp** {*|*address*|*peer-group-name*} **soft out**

Example:

```
Device# clear ip bgp * soft out
```

(Optional) Performs an outbound soft reset to reset the inbound routing table on the specified connection. Use this command if route refresh is supported.

- Enter an asterisk (*) to specify that all connections be reset.
- Enter an IP address to specify the connection to be reset.
- Enter a peer group name to reset the peer group.

Step 4 **show ip bgp**

Example:

```
Device# show ip bgp
```

Verifies the reset by checking information about the routing table and about BGP neighbors.

Step 5 **show ip bgp neighbors**

Example:

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

Verifies the reset by checking information about the routing table and about BGP neighbors.

Configuring BGP Decision Attributes

Before you begin

On the Cisco Catalyst 9300 Series Switches, the hardware-supported maximum-path is 8.

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(config)# router bgp 4500
```

Enables a BGP routing process, assign it an AS number, and enter router configuration mode.

Step 4 **bgp best-path as-path ignore**

Example:

```
Device(config-router)# bgp bestpath as-path ignore
```

(Optional) Configures the router to ignore AS path length in selecting a route.

Step 5 **neighbor** [*ip-address* | *peer-group-name*} **next-hop-self**

Example:

```
Device(config-router)# neighbor 10.108.1.1 next-hop-self
```

(Optional) Disables next-hop processing on BGP updates to a neighbor by entering a specific IP address to be used instead of the next-hop address.

Step 6 **neighbor** [*ip-address* | *peer-group-name*} **weight** *weight*

Example:

```
Device(config-router)# neighbor 172.16.12.1 weight 50
```

(Optional) Assign a weight to a neighbor connection. Acceptable values are from 0 to 65535; the largest weight is the preferred route. Routes that are learned through another BGP peer have a default weight of 0; routes that are sourced by the local router have a default weight of 32768.

Step 7 **default-metric***number*

Example:

```
Device(config-router)# default-metric 300
```

(Optional) Sets a MED metric to set preferred paths to external neighbors. All routes without a MED will also be set to this value. The range is 1 to 4294967295. The lowest value is the most desirable.

Step 8 **bgp bestpath med missing-as-worst**

Example:

```
Device(config-router)# bgp bestpath med missing-as-worst
```

(Optional) Configures the switch to consider a missing MED as having a value of infinity, making the path without a MED value the least desirable path.

Step 9 **bgp always-compare med**

Example:

```
Device(config-router)# bgp always-compare-med
```

(Optional) Configures the switch to compare MEDs for paths from neighbors in different autonomous systems. By default, MED comparison is only done among paths in the same AS.

Step 10 **bgp bestpath med confed**

Example:

```
Device(config-router)# bgp bestpath med confed
```

(Optional) Configures the switch to consider the MED in choosing a path from among those advertised by different subautonomous systems within a confederation.

Step 11 **bgp deterministic med**

Example:

```
Device(config-router)# bgp deterministic med
```

(Optional) Configures the switch to consider the MED variable when choosing among routes advertised by different peers in the same AS.

Step 12 **bgp default local-preference** *value*

Example:

```
Device(config-router)# bgp default local-preference 200
```

(Optional) Change the default local preference value. The range is 0 to 4294967295; the default value is 100. The highest local preference value is preferred.

Step 13 **maximum-paths ibgp** *number*

Example:

```
Device(config-router)# maximum-paths 8
or
Device(config-router)# maximum-paths ibgp 2
```

(Optional) Configures the number of paths to be added to the IP routing table. The default is to only enter the best path in the routing table. Having multiple paths allows load-balancing among the paths.

- number : The number of paths to be added to the IP routing table.

  Note
  Although the switch software allows a maximum of 32 equal-cost routes, the actual number of paths per route is dependent on the switch hardware.

- ibgp number : The number of iBGP paths to be added to the IP routing table.

  Note
  The **maximum-paths ibgp** *number* command is supported from Cisco IOS XE Cupertino 17.9.6 and all subsequent Cisco IOS XE Cupertino 17.9.x releases, and from Cisco IOS XE 17.15.1 and all subsequent releases.

Step 14 **end**

Example:

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

Returns to privileged EXEC mode.

Step 15 **show ip bgp**

Example:

```
Device# show ip bgp
```

Verifies the reset by checking information about the routing table and about BGP neighbors.

Step 16 **show ip bgp neighbors**

Example:

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

Verifies the reset by checking information about the routing table and about BGP neighbors.

Step 17 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Configuring BGP Filtering with Route Maps

•

Before you begin

•

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **route-map** *map-tag* [**permit**|**deny**][*sequence-number*]

Example:

```
Device(config)# route-map set-peer-address permit 10
```

Creates a route map, and enter route-map configuration mode.

Step 4 **set ip next-hop** *ip-address* [*...ip-address*][**peer-address**]

Example:

```
Device(config)# set ip next-hop 10.1.1.3
```

(Optional) Sets a route map to disable next-hop processing

- In an inbound route map, set the next hop of matching routes to be the neighbor peering address, overriding third-party next hops.

- In an outbound route map of a BGP peer, set the next hop to the peering address of the local router, disabling the next-hop calculation.

Step 5 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 6 **show route-map** [*map-name*]

Example:

```
Device# show route-map
```

Displays all route maps configured or only the one specified to verify configuration.

Step 7 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Configuring BGP Filtering by Access Lists and Neighbors

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip as-path access-list** *access-list-number* {**permit**|**deny**} *as-regular-expressions*

Example:

```
Device(config)# ip as-path access-list 1 deny _65535_
```

Defines a BGP-related access list.

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

Example:

```
Device(config)# router bgp 110
```

Enters BGP router configuration mode.

Step 5 **neighbor** {*ip-address* | *peer-group name*} **filter-list** {*access-list-number* | *name*} {**in** | **out** | **weight** *weight*}

Example:

```
Device(config-router)# neighbor 172.16.1.1 filter-list 1 out
```

Establishes a BGP filter based on an access list.

Step 6 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 7 **show ip bgp neighbors** [**paths** *regular-expression*]

Example:

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

Verifies the configuration.

Step 8 **copy running-config startup-config**[**paths** *regular-expression*]

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Configuring Prefix Lists for BGP Filtering

You do not need to specify a sequence number when removing a configuration entry. Show commands include the sequence numbers in their output. Before using a prefix list in a command, you must set up the prefix list. To configure prefix list for BGP filtering, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip prefix-list** *list-name*[**seq** *seq-value*]**deny**|**permit** *network/len*[**ge** *ge-value*][**le** *e-value*]

Example:

```
Device(config)# ip prefix-list BLUE permit 172.16.1.0/24
```

Creates a prefix list with an optional sequence number to **deny**or **permit** access for matching conditions. You must enter at least one **permit** or **deny** clause.

- *network/len* is the network number and length (in bits) of the network mask.
- Optional) **ge** and **le** values specify the range of the prefix length to be matched. The specified *ge-value* and *le-value* must satisfy this condition: *len* < *ge-value* < *le-value* < *32*

Step 4 **ip prefix-list** *list-name*[**seq** *seq-value*]**deny**|**permit** *network/len*[**ge** *ge-value*][**le** *e-value*]

Example:

```
Device(config)# ip prefix-list BLUE seq 10 permit 172.24.1.0/24
```

(Optional) Adds an entry to a prefix list, and assign a sequence number to the entry.

Step 5 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 6 **show ip prefix list** [**detail**|**summary**] *name* [*network/len*] [**seq seq-num**] [**longer**] [**first-match**]

Example:

```
Device# show ip prefix list summary test
```

Verifies the configuration by displaying information about a prefix list or prefix list entries.

Step 7 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Configuring BGP Community Filtering

By default, no COMMUNITIES attribute is sent to a neighbor. You can specify that the COMMUNITIES attribute be sent to the neighbor at an IP address by using the **neighbor send-community** router configuration command.

To configure BGP community filter, perform this procedure:

Before you begin

-

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **ip community-list** *community-list-number*[**permit**|**deny**]| *community-number*[**ge** *ge-value*

Example:

```
Device(config)# ip community-list 1 permit 50000:10
```

Creates a community list, and assigns it a number.

- The *community-list-number* is an integer from 1 to 99 that identifies one or more permit or deny groups of communities.
- The *community-number* is the number that is configured by a **set community** route-map configuration command.

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

Example:

```
Device(config)# router bgp 108
```

Enters BGP router configuration mode.

Step 5 **neighbor** {*ip-address*|*peer-group name*}**send-community**

Example:

```
Device(config-router)# neighbor 172.16.70.23 send-community
```

Specifies that the COMMUNITIES attribute be sent to the neighbor at this IP address.

Step 6 **set comm-list** *list-num* **delete**

Example:

```
Device(config-router)# set comm-list 500 delete
```

(Optional) Removes communities from the community attribute of an inbound or outbound update that match a standard or extended community list that is specified by a route map.

Step 7 **exit**

Example:

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

Returns to global configuration mode.

Step 8 **ip bgp-community new-format**

Example:

```
Device(config)# ip bgp-community new format
```

(Optional) Displays and parses BGP communities in the format AA:NN. A BGP community is displayed in a two-part format 2 bytes long. The Cisco default community format is in the format NNAA. In the most recent RFC for BGP, a community takes the form AA:NN, where the first part is the AS number and the second part is a 2-byte number.

Step 9 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 10 **show ip bgp community**

Example:

```
Device# show ip bgp community
```

Verifies the configuration.

Step 11 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Configuring BGP Neighbors and Peer Groups

To assign configuration options to an individual neighbor, specify any of these router configuration commands by using the neighbor IP address. To assign the options to a peer group, specify any of the commands by using the peer group name. You can disable a BGP peer or peer group without removing all the configuration information by using the **neighbor shutdown** router configuration command.

To configure BGP neighbors and peer groups, perform this procedure:

Before you begin

-

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

Enters BGP router configuration mode.

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

Example:

Creates a BGP peer group.

Step 5 **neighbor** *ip-address* **peer-group***peer-group-name*

Example:

Makes a BGP neighbor a member of the peer group.

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

Example:

Specifies a BGP neighbor. If a peer group is not configured with a **remote-as** *number* , use this command to create peer groups containing EBGP neighbors. The range is 1 to 65535.

Step 7 **neighbor** {*ip-address* |*peer-group-name*} **description** *text*

Example:

(Optional) Associates a description with a neighbor.

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

Example:

(Optional) Allows a BGP speaker (the local router) to send the default route 0.0.0.0 to a neighbor for use as a default route.

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

Example:

(Optional) Specifies that the COMMUNITIES attribute be sent to the neighbor at this IP address.

Step 10 **neighbor** {*ip-address* |*peer-group-name*} **update-source** *interface*

Example:

(Optional) Allows internal BGP sessions to use any operational interface for TCP connections.

Step 11 **neighbor** {*ip-address* |*peer-group-name*} **ebgp-multihop**

Example:

(Optional) Allows BGP sessions, even when the neighbor is not on a directly connected segment. The multihop session is not established if the only route to the multihop peer's address is the default route (0.0.0.0).

Step 12 **neighbor** {*ip-address* |*peer-group-name*} **local-as** *number*

Example:

(Optional) Specifies an AS number to use as the local AS. The range is 1 to 65535.

Step 13 **neighbor** {*ip-address* |*peer-group-name*} **advertisement-interval** *seconds*

Example:

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

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

Example:

(Optional) Controls how many prefixes can be received from a neighbor. The range is 1 to 4294967295. The threshold (optional) is the percentage of maximum at which a warning message is generated. The default is 75 percent.

Step 15 **neighbor** {*ip-address* |*peer-group-name*} **next-hop-self**

Example:

(Optional) Disables next-hop processing on the BGP updates to a neighbor.

Step 16 **neighbor** {*ip-address* |*peer-group-name*} **password** *{0-7}string*

Example:

(Optional) Sets MD5 authentication on a TCP connection to a BGP peer. The same password must be configured on both BGP peers, or the connection between them is not made.

The encryption modes supported for the password are:

- 0 - no encryption/plaintext
- 7 - proprietary encryption type

Type 7 is used only for storing the password in the device configuration. The actual value that gets used at the time of BGP session establishment is the MD5 hash of the plaintext password.

- *string* - the password string

Step 17 **neighbor** {*ip-address* |*peer-group-name*} **route-map** *map-name*{**in**|**out**}

Example:

(Optional) Applies a route map to incoming or outgoing routes.

Step 18 **neighbor** {*ip-address* |*peer-group-name*} **send-community**

Example:

(Optional) Specifies that the COMMUNITIES attribute be sent to the neighbor at this IP address.

Step 19 **neighbor** {*ip-address* |*peer-group-name*} **timers** *keepalive holdtime*

Example:

(Optional) Sets timers for the neighbor or peer group.

- The *keepalive* interval is the time within which keepalive messages are sent to peers. The range is 1 to 4294967295 seconds; the default is 60.
- The *holdtime* is the interval after which a peer is declared inactive after not receiving a keepalive message from it. The range is 1 to 4294967295 seconds; the default is 180.

Step 20 **neighbor** {*ip-address* |*peer-group-name*} **weight** *weight*

Example:

(Optional) Specifies a weight for all routes from a neighbor.

Step 21 **neighbor** {*ip-address* |*peer-group-name*} **distribute-list** { *access-list-number*|*name*} {**in**|**out**}

Example:

(Optional) Filter BGP routing updates to or from neighbors, as specified in an access list.

Step 22 **neighbor** {*ip-address* |*peer-group-name*} **filter-list** { *access-list-number* {**in**|**out**|**weight** *weight*}

Example:

(Optional) Establish a BGP filter.

Step 23 **neighbor** {*ip-address* |*peer-group-name*} **version** *value*

Example:

(Optional) Specifies the BGP version to use when communicating with a neighbor.

Step 24 **neighbor** {*ip-address* |*peer-group-name*} **soft-reconfiguration inbound**

Example:

(Optional) Configures the software to start storing received updates.

Step 25 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 26 **show ip bgp neighbors**

Example:

```
Device(config)# end
```

Verifies the configuration.

Step 27 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Configuring Aggregate Addresses in a Routing Table

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(config)# router bgp 106
```

Enters BGP router configuration mode.

Step 4 **aggregate-address** *address-mask*

Example:

```
Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0
```

Creates an aggregate entry in the BGP routing table. The aggregate route is advertised as coming from the AS, and the atomic aggregate attribute is set to indicate that information might be missing.

Step 5 **aggregate-address** *address-mask* **as-set**

Example:

```
Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 as-set
```

(Optional) Generates AS set path information. This command creates an aggregate entry following the same rules as the previous command, but the advertised path will be an AS_SET consisting of all elements contained in all paths. Do not use this keyword when aggregating many paths because this route must be continually withdrawn and updated.

Step 6 **aggregate-address** *address-mask* **summary-only**

Example:

```
Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 summary-only
```

(Optional) Advertises summary addresses only.

Step 7 **aggregate-address** *address-mask* **suppress-map** *map-name*

Example:

```
Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 suppress-map map1
```

(Optional) Suppresses selected, more specific routes.

Step 8 **aggregate-address** *address-mask* **advertise-map** *map-name*

Example:

```
Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 advertise-map map2
```

(Optional) Generates an aggregate based on conditions that are specified by the route map.

Step 9 **aggregate-address** *address-mask* **attribute-map** *map-name*

Example:

```
Device(config-router)# aggregate-address 10.0.0.0 255.0.0.0 attribute-map map3
```

(Optional) Generates an aggregate with attributes that are specified in the route map.

Step 10 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 11 **show ip bgp neighbors** [**advertised-routes**]

Example:

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

Verifies the configuration.

Step 12 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Example

What to do next

-

Configuring Routing Domain Confederations

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

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

Enters BGP router configuration mode.

Step 4 **bgp confederation identifier** *autonomous-system*

Example:

```
Device(config)# bgp confederation identifier 50007
```

Configures a BGP confederation identifier.

Step 5 **bgp confederation identifier** *autonomous-system*[*autonomous-system...*]

Example:

```
Device(config)# bgp confederation peers 51000 51001 51002
```

Specifies the autonomous systems that belong to the confederation and that will be treated as special EBGP peers.

Step 6 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 7 **show ip bgp neighbor**

Example:

```
Device# show ip bgp neighbor
```

Verifies the configuration.

Step 8 **show ip bgp network**

Example:

```
Device# show ip bgp network
```

Verifies the configuration.

Step 9 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Example

What to do next

-

Configuring BGP Route Reflectors

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(config)# router bgp 101
```

Enters BGP router configuration mode.

Step 4 **neighbor**{ *ip-address*|*peer-group-name*}**route-reflector-client**

Example:

```
Device(config-router)# neighbor 172.16.70.24 route-reflector-client
```

Configures the local router as a BGP route reflector and the specified neighbor as a client.

Step 5 **bgp cluster-id** *cluster-id*

Example:

```
Device(config-router)# bgp cluster-id 10.0.1.2
```

(Optional) Configures the cluster ID if the cluster has more than one route reflector.

Step 6 **no bgp client-to-client reflection**

Example:

```
Device(config-router)# no bgp client-to-client reflection
```

(Optional) Disables client-to-client route reflection. By default, the routes from a route reflector client are reflected to other clients. However, if the clients are fully meshed, the route reflector does not need to reflect routes to clients.

Step 7 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 8 **show ip bgp**

Example:

```
Device# show ip bgp
```

Verifies the configuration. Displays the originator ID and the cluster-list attributes.

Step 9 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Example

What to do next

-

Configuring Route Dampening

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

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

Enters BGP router configuration mode.

Step 4 **bgp dampening**

Example:

```
Device(config-router)# bgp dampening
```

Enables BGP route dampening.

Step 5 **bgp dampening***half-life reuse suppress max-suppress*[**route-map** *map*]

Example:

```
Device(config-router)# bgp dampening 30 1500 10000 120
```

(Optional) Changes the default values of route dampening factors.

Step 6 **end**

Example:

```
Device(config)# end
```

Returns to privileged EXEC mode.

Step 7 **show ip bgp flap-statistics** [{**regexp** *regexp*} | {**filter-list** *list*} | {*address mask*[**longer-prefix**]}]

Example:

```
Device# show ip bgp flap-statistics
```

(Optional) Monitors the flaps of all paths that are flapping. The statistics are deleted when the route is not suppressed and is stable.

Step 8 **show ip bgp dampened-paths**

Example:

```
Device# show pi bgp dampened-paths
```

(Optional) Displays the dampened routes, including the time remaining before they are suppressed.

Step 9 **clear ip bgp flap-statistics** [{**regexp** *regexp*} | {**filter-list** *list*} | {*address mask*[**longer-prefix**]}]

Example:

```
Device# clear ip bgp flap-statistics
```

(Optional) Clears BGP flap statistics to make it less likely that a route will be dampened.

Step 10 **clear ip bgp dampening**

Example:

```
Device# clear ip bgp dampening
```

(Optional) Clears route dampening information, and unsuppress the suppressed routes.

Step 11 **copy running-config startup-config**

Example:

```
Device# copy running-config startup-config
```

(Optional) Saves your entries in the configuration file.

Conditionally Injecting BGP Routes

Before you begin

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

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(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:

```
Device(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 inherits the attributes of the aggregate route.

Step 5 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

Step 6 **route-map** *map-tag*[**permit**|**deny**][*sequence-number*]

Example:

```
Device(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-last-name*[*prefix-list-name...*]}

Example:

```
Device(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 that is 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:

```
Device(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 that is 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.

Step 9 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

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

Example:

```
Device(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-last-name*[*prefix-list-name...*]}

Example:

```
Device(config-route-map)# set ip address prefix-list ORIGINATED_ROUTES
```

Specifies the routes to be injected. In this example, the prefix list that is named originated_routes is used to redistribute the source of the route.

Step 12 **set community**{*community-number*[**additive**][*well-known-community*]|**none**}

Example:

```
Device(config-route-map)# set community 14616:555 additive
```

Sets the BGP community attribute of the injected route.

Step 13 **exit**

Example:

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

Exits router 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:

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

Configures a prefix list. In this example, the prefix list that is named SOURCE is configured to permit routes from network 10.1.1.0/24.

Step 15 **Example:**

Repeat Step 14 for every prefix list to be created.

Step 16 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

Step 17 **show ip bgp injected-paths**

Example:

```
Device# show ip bgp injected-paths
```

(Optional) Displays information about injected paths.

Configuring Peer Session Templates

The following example creates a peer session template that is 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
```

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.

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

To configure a basic peer session template, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp autonomous-system- number**

Example:

```
Device(config)# router bgp
101
```

Enters router configuration mode and creates a BGP routing process.

Step 4 **template peer-session sessiontemplate-name**

Example:

```
Device(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:

```
Device(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 holdtime**

Example:

```
Device(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:

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

Exits session-template configuration mode and returns to privileged EXEC mode.

Step 8 **show ip bgp template peersession [session-template- name]**

Example:

```
Device# 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 sessiontemplate-name argument. This command also supports all standard output modifiers.

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



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

To configure peer session template inheritance, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(config)# router bgp 101
```

Enters router configuration mode and creates a BGP routing process.

Step 4 **description***text-string*

Example:

```
Device(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 5 **update-source** *interface-type interface-number*

Example:

```
Device(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 6 **inherit peer-session** *session-template-name*

Example:

```
Device(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 7 **end**

Example:

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

Returns to privileged EXEC mode.

Step 8 **show ip bgp template peer-session** [*session-template-name*]

Example:

```
Device# 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.

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

To configure peer session template inheritance, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(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:

```
Device(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*

Example:

```
Device(config-router)# neighbor 172.16.0.1 inherit peer-session CORE1
```

Sends a peer session template to a neighbor so that the neighbor can inherit the configuration.

The example configures a device 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:

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

Returns to privileged EXEC mode.

Step 7 **show ip bgp template peer-session** [*session-template-name*]

Example:

```
Device# 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.

Configuring Peer Policy Templates

Procedure

Example:

Configuring Basic Peer Policy Templates



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.

To configure a basic peer policy template, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(config)# router bgp 101
```

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 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.

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
```

Returns to privileged EXEC mode.

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

To configure peer policy template inheritance, perform this procedure:

Before you begin

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

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

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 NETWORK1
```

Enter policy-template configuration mode and creates a peer policy template.

Step 5 **route-map** *map-name* {**in**|**out** }

Example:

```
Device(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.

Step 6 **inherit peer-policy** *policy-template-name sequence-number*

Example:

```
Device(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.

Step 7 **end**

Example:

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

Returns to privileged EXEC mode.

Step 8 **show ip bgp template peer-policy** [*policy-template-name*[**detail**]]

Example:

```
Device# 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.

Example

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.

```
Device# 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

To configure peer policy template, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *autonomous-system*

Example:

```
Device(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:

```
Device(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.

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

Example:

```
Device(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-session***policy-template-name*

Example:

```
Device(config-router-af)# neighbor 192.168.1.2 inherit peer-policy GLOBAL
```

Sends a peer session template to a neighbor so that the neighbor can inherit the configuration.

The example configures a router to send the peer policy template that is 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:

```
Device(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:

```
Device# 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 that are applied to this neighbor per address family.
- Use the **detail** keyword to display detailed policy information.

Example

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

```
Device# 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
```

Configuring BGP Route Map Next-hop Self

To configure BGP route map next-hop self, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **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:

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

Exits route-map configuration mode and enters global configuration mode.

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* |*ipv6-address*|*peer-group-name*} **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.

Step 15 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **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* | *ipv6-address* | *peer-group-name*} **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.

- *string* - the password string

Step 17 **neighbor** {*ip-address* | *ipv6-address* | *peer-group-name*} **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** *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.

Step 24 **end**

Example:

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

Exits router configuration mode and enters privileged EXEC mode.

Configuration Examples for BGP

Configuring Conditional BGP Route Injection

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

```
Device# 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 ?
```

Configuring Peer Session Templates

The following example creates a peer session template that is 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
```

Configuring Peer Policy Templates

The following example creates a peer policy template that is 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 that is 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 that are 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 that are named PRIMARY-IN and GLOBAL, the 192.168.2.2 neighbor will also indirectly inherit the peer policy templates that are 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
```

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
```

Monitoring and Maintaining BGP

To effectively monitor and maintain Border Gateway Protocol (BGP) operations, network administrators have access to various commands that allow for clearing and displaying BGP-related information. This functionality is crucial for troubleshooting, optimizing resource utilization, and ensuring network stability.

Administrators can clear the contents of specific BGP caches, tables, or databases when data integrity is suspected to be compromised. This helps in resetting the BGP state and resolving potential issues.

Furthermore, detailed statistics can be displayed, including the contents of BGP routing tables, caches, and databases. This information provides insights into resource consumption, aids in diagnosing network problems, and helps in understanding node reachability and packet forwarding paths.

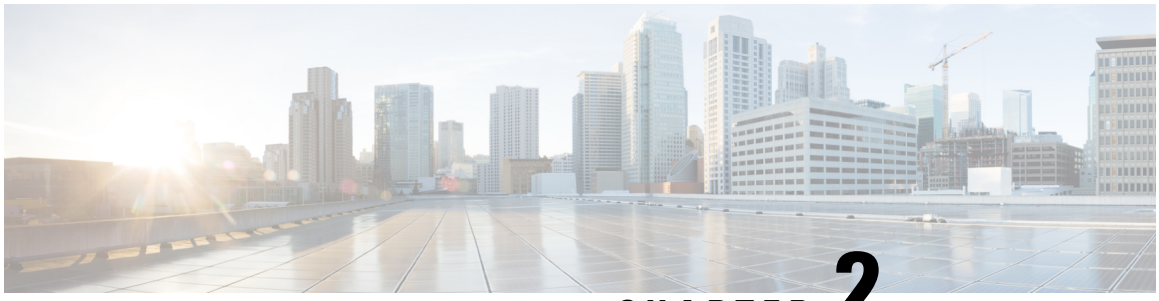
Workflow

The following table lists privileged EXEC commands for clearing and displaying BGP information:

1.	**clear ip bgp address** *address*	Resets a particular BGP connection.
	clear ip bgp *	Resets all BGP connections.
	clear ip bgp peer-group *tag*	Removes all members of a BGP peer group.

show ip bgp *prefix*	Displays peer groups and peers not in peer groups to which the prefix has been advertised. Also displays prefix attributes such as the next hop and the local prefix.
show ip bgp cidr-only	Displays all BGP routes that contain subnet and supernet network masks.
show ip bgp community [*community-number*] [**exact**]	Displays routes that belong to the specified communities.
show ip bgp community *community-number* **exact-match**]	Displays routes that are permitted by the community list.
show ip bgp filter-list *access-list-number*	Displays routes that are matched by the specified AS path access list.
show ip bgp inconsistent-as	Displays the routes with inconsistent originating autonomous systems.
show ip bgp regexp *regular-expression*	Displays the routes that have an AS path that matches the specified regular expression entered on the command line.
show ip bgp	Displays the contents of the BGP routing table.
show ip bgp neighbors [*address*]	Displays detailed information on the BGP and TCP connections to individual neighbors.
show ip bgp neighbors [*address*] [**advertised-routes** |**dampened-routes** |**flap-statistics** |**paths** *regular-expression*|**received-routes** | **routes**]	Displays routes learned from a particular BGP neighbor.
show ip bgp paths	Displays all BGP paths in the database.
show ip bgp peer-group [*tag*] [**summary**]	Displays information about BGP peer groups.
show ip bgp summary	Displays the status of all BGP connections.

The **bgp log-neighbor changes** command is enabled by default. It allows to log messages that are generated when a BGP neighbor resets, comes up, or goes down.



CHAPTER 2

Configure BGP Graceful Shutdown

- Feature history for BGP graceful shutdown, on page 55
- Information about BGP graceful shutdown, on page 55
- Configure BGP graceful shutdown, on page 58
- Configuration examples for BGP graceful shutdown, on page 64
- Additional references, on page 67

Feature history for BGP graceful shutdown

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature Name and Description	Supported Platform
Cisco IOS XE 26.2.1ea	BGP Graceful Shutdown: BGP Graceful Shutdown feature support has been introduced.	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.1	BGP Graceful Shutdown: 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.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

Information about BGP graceful shutdown

This sections provide information about BGP graceful shutdown.

BGP graceful shutdown benefits

BGP graceful shutdown benefits are network advantages that

- reduce or eliminate the loss of inbound or outbound traffic flows during planned BGP session shutdowns
- minimize traffic loss during the planned shutdown and subsequent reestablishment of sessions, and
- provide fewer lost packets and less time spent reconfiguring devices.

Graceful shutdown operation

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 that are 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.

GSHUT community

A GSHUT community is a well-known community attribute that

- is applied to a neighbor to gracefully shut down the link in an expected number of seconds
- is used in conjunction with the BGP Graceful Shutdown feature, and
- is always sent by the GSHUT initiator.

GSHUT community usage

The GSHUT community attribute is applied to a neighbor specified by the **neighbor shutdown graceful** command.

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.

Configure BGP GSHUT enhancement

Configure BGP GSHUT enhancement to enable graceful shutdown functionality for BGP sessions, allowing for controlled network maintenance operations.

Use this feature when you need to perform maintenance on BGP routers while minimizing traffic disruption. The BGP GSHUT enhancement provides a mechanism to gracefully shutdown BGP sessions with specific timing and community attributes.

Procedure

Step 1 **enable**

Example:

```
Device>enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device#configure terminal
```

Enters global configuration mode.

Step 3 **router BGP** *autonomous-system-number*

Example:

```
Device(config)#router bgp 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.

BGP GSHUT enhancement is now configured and activated. The graceful shutdown functionality is enabled for BGP sessions with the specified timing and community attributes.

Configure BGP graceful shutdown

BGP graceful shutdown provides configurational capabilities for managing BGP session termination.

Summary

BGP graceful shutdown allows network administrators to manage BGP session termination in a controlled fashion, reducing disruption and ensuring seamless rerouting during maintenance or shutdown procedures.

The key components involved in the process are:

- Network administrator: Initiates and manages the BGP graceful shutdown configuration process.
- BGP router: Receives and applies the graceful shutdown configuration settings.

Workflow

The process involves the following stages:

1. The network administrator accesses the BGP router configuration interface.
2. The administrator applies the graceful shutdown settings on the router.
3. The BGP router recognizes the graceful shutdown configuration and signals its peers accordingly.

Result

The BGP session terminates in a controlled manner, allowing neighboring routers to reroute traffic and minimize service interruption during maintenance or shutdown.

Shut down a BGP link gracefully

This task allows you to gracefully shut down a BGP link, providing time for iBGP peers to converge and choose an alternate path before the connection is terminated.

Use this procedure when you need to shut down a BGP link in a controlled manner that minimizes network disruption by allowing proper route convergence.

Before you begin

Follow these steps to shut down a BGP link gracefully:

Procedure

Step 1 **enable**

Example:

```
Device>enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device#configure terminal
```

Enters global configuration mode.

Step 3 **router BGP** *autonomous-system-number*

Example:

```
Device(config)#router bgp 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.
- 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.

The BGP link is configured for graceful SHUTDOWN with the specified timer, community values, and local preference settings. The link will shut down gracefully after the configured time period, allowing iBGP peers to converge on alternate paths.

Filter BGP routes based on the GSHUT community

Filter BGP routes that have the GSHUT community to control route acceptance from peers during graceful shutdown scenarios.

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

Before you begin

Follow these steps to filter BGP routes based on the GSHUT community:

Procedure

Step 1 **enable**

Example:

```
Device>enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device#configure terminal
```

Enters global configuration mode.

Step 3 **router BGP** *autonomous-system-number*

Example:

```
Device(config)#router bgp 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.

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 that is named RM_GSHUT denies routes from the specified neighbor that have the GSHUT community.

The route map is configured to deny incoming BGP routes that have the GSHUT community from the specified neighbor, effectively filtering them from the routing table.

Configure BGP GSHUT enhancement

Configure BGP GSHUT enhancement to enable graceful shutdown functionality for BGP sessions, allowing for controlled network maintenance operations.

Use this feature when you need to perform maintenance on BGP routers while minimizing traffic disruption. The BGP GSHUT enhancement provides a mechanism to gracefully shutdown BGP sessions with specific timing and community attributes.

Procedure

Step 1 **enable**

Example:

```
Device>enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device#configure terminal
```

Enters global configuration mode.

Step 3 **router BGP** *autonomous-system-number*

Example:

```
Device(config)#router bgp 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.

BGP GSHUT enhancement is now configured and activated. The graceful shutdown functionality is enabled for BGP sessions with the specified timing and community attributes.

Configuration examples for BGP graceful shutdown

These sections provide configuration examples for BGP graceful shutdown.

Example: shut down a BGP link gracefully

This reference provides examples for 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
```

Filter BGP routes based on the GSHUT community

Filter routes with the GSHUT community from getting into the BGP routing table using a community list and route map configuration.

In addition 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 configuration uses a route map that denies routes based on a standard community list containing routes with the GSHUT community. The route map is then applied to incoming routes from a specific neighbor.

Procedure

Step 1 Configure the BGP router process and neighbor settings.

Example:

```
Device(config)#router bgp 2000
Device(config-router)#neighbor 2001:db8:4::1 remote-as 1000
Device(config-router)#neighbor 2001:db8:4::1 activate
Device(config-router)#neighbor 2001:db8:4::1 send-community
Device(config-router)#exit
```

Step 2 Create a route map that denies routes based on the community list.

Example:

```
Device(config)#route-map RM_GSHUT deny 10
Device(config-route-map)#match community GSHUT
Device(config-route-map)#exit
```

Step 3 Create a standard community list for the GSHUT community.

Example:

```
Device(config)#ip community-list standard GSHUT permit gshut
```

Step 4 Apply the route map to incoming routes from the neighbor.

Example:

```
Device(config)#router bgp 2000
Device(config)#neighbor 2001:db8:4::1 route-map RM_GSHUT in
```

The route map RM_GSHUT filters incoming BGP routes from neighbor 2001:db8:4::1, denying routes that contain the GSHUT community from entering the BGP routing table.

Configure BGP GSHUT enhancement

Enable and activate the BGP GSHUT enhancement feature to configure neighbors for graceful shutdown within a specified time duration.

The BGP GSHUT enhancement allows you to gracefully shutdown BGP neighbors with specific parameters such as local preference and community values. This example demonstrates configuration for multiple neighbor types including IPv4 and VRF neighbors.

Procedure

Step 1 Enable privileged EXEC mode and enter global configuration mode.

Example:

```
Device>enable
Device#configure terminal
```

Step 2 Configure BGP router mode and enable graceful shutdown for all neighbors.

Example:

```
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
```

In this configuration, neighbors are set to gracefully shutdown within 180 seconds with local-preference set to 20 and community value set to 10.

Step 3 Verify the graceful shutdown configuration for specific neighbors.

Example:

```
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
```

This example shows two IPv4 neighbors configured with IP addresses 10.2.2.2 and 172.16.2.1, and one VRF neighbor tagged v1 with IP address 192.168.1.1.

Step 4 Display the running configuration to verify BGP session information.

Example:

```
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
```

The BGP GSHUT enhancement is successfully configured and activated. Neighbors will gracefully shutdown within the specified 180-second duration with the configured local preference and community values. The configuration can be verified using the show commands to display graceful shutdown timers and running configuration details.

Additional references

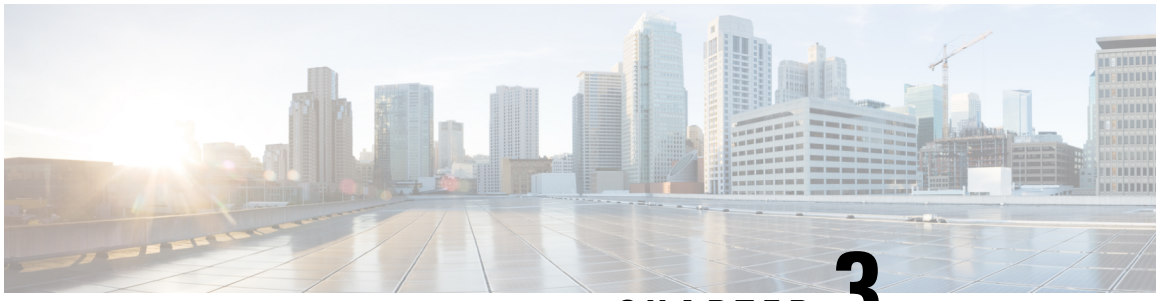
This reference provides additional documentation and standards related to BGP graceful shutdown implementation.

Related documents

Related Topic	Document Title
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*



CHAPTER 3

Configure BGP Large Community

- Feature history for BGP large community, on page 69
- Restrictions for the BGP large community, on page 70
- BGP large community feature, on page 70
- Configure BGP large community, on page 71
- Configuration example for BGP large community, on page 79

Feature history for BGP large community

This table provides release and related information for the features explained in this module.

These features are available on all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature Information
Cisco IOS XE 26.2.1ea	BGP Large Community: BGP Large Community feature support has been introduced	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.1	BGP Large Community: The BGP large communities attribute provides the capability for tagging routes and modifying BGP routing policy on routers. They are similar attributes to BGP communities, but with a twelve octet size.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/.

Restrictions for the BGP large community

When large communities are specified in commands, they are specified as three non-negative decimal integers separated by colons. For example as 1:2:3. Each integer is stored in 32 bits. The possible range for each integer is four octet decimal which can be from 0 to 4294967295.

BGP large community feature

A BGP large community feature is a routing attribute that

- provides the capability for tagging routes and modifying BGP routing policy on routers
- allows BGP large communities to be appended or removed selectively on the large community attribute as the route travels from router to router
- uses a twelve octet size similar to BGP communities but without well-known large communities.

BGP large community structure

The BGP large communities are split logically into a 4 octet Global Administrator field and a 8 octet Local Administrator field. A 4 octet Autonomous System can fit into the Global Administrator field.

For more information on BGP large community, see the rfc8092 document.

Large community lists

A large community list is a BGP routing mechanism that

- creates groups of large communities which can be used in a match clause of a route map
- controls routing policy by allowing you to filter the routes you receive or advertise, or modify the attributes of the routes you receive or advertise, and
- sets or deletes the large communities selectively.

Large community list types and characteristics

Large community lists have two types:

- Standard large community lists are used to specify large communities.
- Expanded large community lists are used to specify large communities using a regular expression.

A large community list can be either named or numbered and standard or expanded. All the rules of numbered large community lists apply to named large community lists, except that there is no limit on the number of named community lists that can be configured.



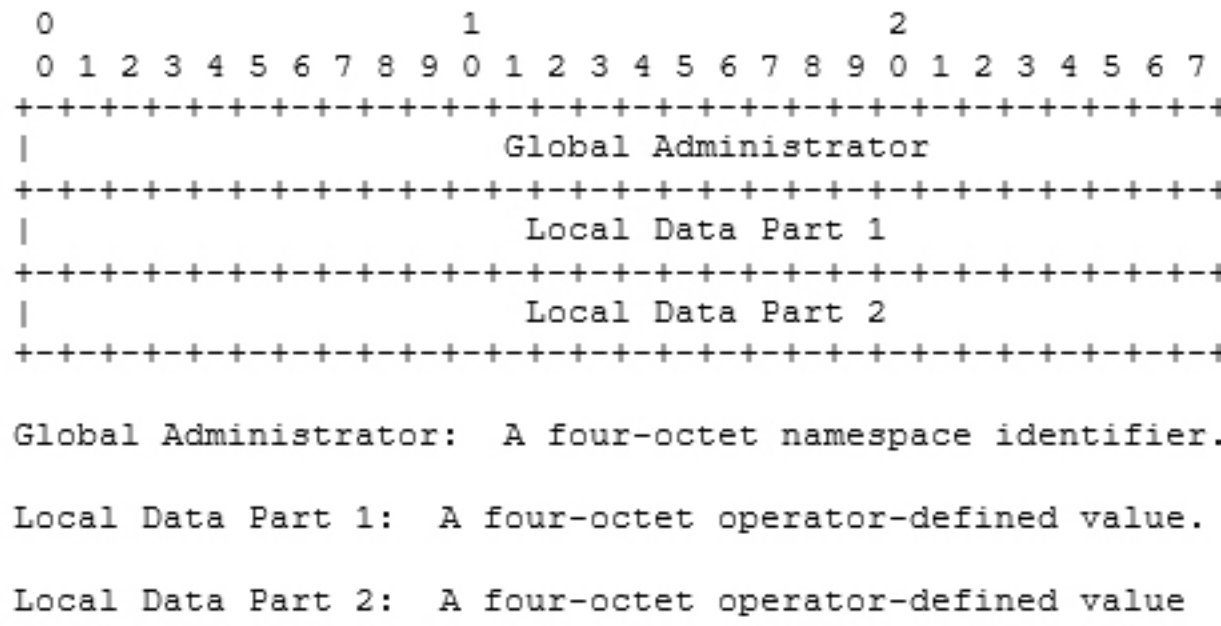
Note A maximum of 100 numbered standard large community lists and 100 numbered expanded large community lists can be configured. A named large community list does not have this limitation.

BGP large community attribute

A BGP large community attribute is a BGP attribute that encodes the community value as a 12 octet number.

Large community attribute syntax

This image displays the syntax of the large community attribute.



Configure BGP large community

These sections provide configuration information about BGP large community.

Enable BGP large community

Enable BGP large community support to allow the router to send and receive community attributes with BGP neighbors, providing enhanced routing policy control and traffic engineering capabilities.

BGP large community attributes provide a mechanism for grouping destinations and applying routing policies. This configuration enables the router to exchange community information with BGP peers for improved network management and policy implementation.

Before you begin

Follow these steps to enable BGP large community:

Procedure

Step 1 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 2 **router BGP** *autonomous-system-number*

Example:

```
Device(config)# router bgp 64496
```

Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number.

Step 3 **neighbor** *IP address* **remote-AS** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 209.165.201.1 remote-as 100
```

Enters global address family configuration mode. This command triggers an automatic notification and session reset for all BGP neighbors.

Step 4 **address-family** { **ipv4 | ipv6 | l2vpn | nsap** {**unicast |multicast** }}

Example:

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

Enters global address family configuration mode. This command triggers an automatic notification and session reset for all BGP neighbors.

Note
It also supports other available address families.

Step 5 **neighbor** *IP address* **activate**

Example:

```
Device(config-router)# neighbor 209.165.201.1 activate
```

Enters global address family configuration mode and activates the BGP neighbor.

Step 6 **neighbor** *IP address* **send-community** {**both | extended | standard**}

Example:

```
Device(config-router-neighbor-af)# neighbor 209.165.201.1 send-community standard
```

Configures the router to send the large-community attribute to the neighbor 209.165.201.1.

- Both—Sends both the extended large community and standard large community attributes to the neighbor.

- Extended—Sends the extended community attribute to the neighbor.
- Standard—Sends large community and also (regular) community attribute to the neighbor.

Step 7 **exit**

Example:

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

Exits address-family mode and router configuration mode and enters global configuration mode.

Step 8 **end**

Example:

```
Device(config)# end
```

Exits configuration mode and enters privileged EXEC mode.

BGP large community is now enabled and configured to exchange community attributes with the specified neighbor. The router can send and receive community information for enhanced routing policy control.

Configure route-map with large community lists and matching a large community

This task enables you to configure a route-map that can match BGP large communities using large community lists, providing flexible routing policy control for BGP routes.

Route-maps with large community matching allow you to filter and control BGP routes based on large community attributes, which provide extended community functionality with a larger address space than standard communities.

Before you begin

Follow these steps to match a BGP large community:

Procedure

Step 1 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 2 **route-map** *map-tag* **[permit | deny]** *[sequence number]*

Example:

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

Enters the route-map configuration mode and defines the conditions for routes from one routing protocol into another.

Step 3 **match large-community** {*name* | *numbered* }

Example:

```
Device(config-route-map)# match large-community 1
```

Matches a large-community list. Defines the rules for an entry in the large-community lists and ensures that all the large communties match the large communities in the routes.

Step 4 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

Step 5 **route-map** *map-tag* **[permit | deny]** *[sequence number]*

Example:

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

Enters the route-map configuration mode and defines the conditions for routes from one routing protocol into another.

Step 6 **match large-community** {*name* | *numbered* } **exact match**

Example:

```
Device(config-route-map)# match large-community 1 exact-match
```

Matches a large-community list. Defines the rules for an entry in the large-community lists and ensures that all the large communties match the large communities in the routes. The key word exact-match indicates that an exact match is required to match a BGP large community.

Step 7 **end**

Example:

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

Exits route map configuration mode and enters privileged EXEC mode.

The route-map is configured to match BGP large communities using the specified large community list. The route-map can now be applied to BGP routing policies to filter routes based on large community attributes.

Define BGP large community list

Define BGP large community lists to control routing policies based on large community attributes.

BGP large community supports named and numbered community lists that can be used for routing policy control.

Before you begin

Follow these steps to define the BGP large community list:

Procedure

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 **ip large-community-list** {*standard-list-number* | **standard** *standard-list-name* } {**deny** | **permit**} *community-number large-community*

Example:

Numbered Large-community List

```
ip large-community-list 1 permit 1:2:3 5:6:7
ip large-community-list 1 permit 4123456789:4123456780:4123456788
```

Named Large-community List

```
ip large-community-list standard LG_ST permit 1:2:3 5:6:7
ip large-community-list standard LG_ST permit 4123456789:4123456780:4123456788
```

Defining the large community based on the standard list number. If you attempt to configure more than 6 communities, the trailing communities that exceed the limit are not processed or saved to the running configuration file.

Step 4 **ip large-community-list** {*expanded-list number* | **expanded** *expanded-list-name*} {**deny** | **permit**} *regexp*

Example:

Numbered Extended Large-community List

```
ip large-community-list 100 permit ^5:.*:7$
ip large-community-list 100 permit ^5:.*:8$
```

Named Extended Large-community List

```
ip large-community-list expanded LG_EX permit ^5:.*:7$
ip large-community-list expanded LG_EX permit ^5:.*:8$
```

Defines the large communities based on regular expression and matches according to Cisco's regular expression implementation.

Step 5 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

Step 6 **end**

Example:

```
Device(config)# end
```

Exits route map configuration mode and enters privileged EXEC mode.

The BGP large community list is defined and ready for use in routing policies.

Configure the route-map to set BGP large communities

This task allows you to configure a route-map that sets BGP large communities on routes, enabling you to apply community-based policies for BGP routing decisions.

Route-maps provide a method for controlling and modifying routing information. Large communities offer an extended community attribute that provides more flexible policy control compared to standard communities.

Before you begin

Follow these steps to configure the route-map to set BGP large communities:

Procedure

Step 1 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 2 **route-map** *map-tag* **[permit | deny]** *[sequence number]*

Example:

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

Enters the route-map configuration mode and specifies a set of large communities to a route.

Step 3 **set large-community** {**none** | {**xx:yy:zz** }}

Example:

```
Device(config-route-map)# set large-community 1:2:3 5:6:7
```

A route-map set statement is used to set large communities in a route. It can specify a set of large communities to a route.

Step 4 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

Step 5 **route-map** *map-tag* **[permit | deny]** *[sequence number]*

Example:

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

Enters the route-map configuration mode and specifies a set of large communities to a route.

Step 6 **set large-community** {**none** | {**xx:yy:zz** | **additive** }}

Example:

```
Device(config-route-map)# set large-community 1:2:3 5:6:7 additive
```

A route-map set statement is used to set large communities in a route. It can specify a set of large communities to a route. Also, the keyword additive adds the large communities without removing the existing large communities.

Step 7 **end**

Example:

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

Exits route map configuration mode and enters privileged EXEC mode.

The route-map is configured to set BGP large communities on matching routes. The large communities can be used for policy-based routing and traffic engineering decisions.

Delete large communities

Delete BGP large communities from routes using route map configuration.

Use this procedure to remove specific large communities from BGP routes based on large-community-list matches.

Before you begin

Follow these steps to delete large communities:

Procedure

Step 1 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 2 **route-map** *map-tag* **[permit | deny]** *[sequence number]*

Example:

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

Enters the route-map configuration mode and defines the conditions for redistributing routes from one routing protocol into another.

Step 3 **set large-comm-list** *community-list-name* **delete**

Example:

```
Device(config-route-map)# set large-comm-list 1 delete
Device(config-route-map)#
```

Deletes the large-communities based on large-community-list matches.

Step 4 **exit**

Example:

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

Exits router configuration mode and enters global configuration mode.

Step 5 **end**

Example:

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

Exits route map configuration mode and enters privileged EXEC mode.

The route map is configured to delete large communities based on the specified large-community-list matches.

Verify the configuration of the BGP large community

To verify the BGP large community, use the show BGP large-community command. This example shows a list of routes that contain all of the large communities that are given in the command. The listed routes may contain additional large communities.

Use this command to display routes containing the specified large communities:

```
Device# show bgp large-community 1:2:3 5:6:7
BGP table version is 17, local router ID is 1.1.1.3
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
 *>i 5.5.5.5/32       1.1.1.2                  0    100      0 ?
 *>i 5.5.5.6/32       1.1.1.2                  0    100      0 ?
```

This example displays the listed routes that contain only the given large communities when you add the keyword exact-match in configuration.

```
Device#show bgp large-community 1:2:3 5:6:7 exact-match
BGP table version is 17, local router ID is 1.1.1.3
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
 *>i 5.5.5.5/32       1.1.1.2                  0    100      0 ?
```

In these examples, the routes 5.5.5.5/32 and 5.5.5.6/32 contain both the large communities 1:2:3 and 5:6:7. The route 5.5.5.6/32 contains some additional large communities.

This example displays a large-community list.

```
Device#show ip largecommunity-list 20
Large Community standard list 20
     permit 1:1:2

Device#show bgp large-community-list 20
Large Community standard list 20
     permit 1:1:2
```

Troubleshoot large communities

To debug the large communities, use **debug ip bgp update** command.

```
Device#debug ip bgp update

*Mar 10 23:25:01.194: BGP(0): 192.0.0.1 rcvd UPDATE w/ attr: nexthop 192.0.0.1, origin ?,
metric 0, merged path 1, AS_PATH , community 0:44 1:1 2:3, large-community 3:1:244 3:1:245
*Mar 10 23:25:01.194: BGP(0): 192.0.0.1 rcvd 5.5.5.1/32
*Mar 10 23:25:01.194: BGP(0): Revise route installing 1 of 1 routes for 5.5.5.1/32 ->
192.0.0.1(global) to main IP table
```

Memory Display

The **show ip bgp summary** command displays large-community memory information.

```
Device #show ip bgp summary
BGP router identifier 1.1.1.1, local AS number 1
BGP table version is 3, main routing table version 3
2 network entries using 496 bytes of memory
2 path entries using 272 bytes of memory
1/1 BGP path/bestpath attribute entries using 288 bytes of memory
1 BGP community entries using 40 bytes of memory
2 BGP large-community 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 using 1096 total bytes of memory
BGP activity 3/1 prefixes, 3/1 paths, scan interval 60 secs
2 networks peaked at 13:04:52 Mar 11 2020 EST (00:07:25.579 ago)

Neighbor        V           AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
192.0.0.2       4            2    1245    1245        3    0    0 18:47:56        0
```

Configuration example for BGP large community

This example shows how to configure route-maps using large-communities.

Setting large communities

A route-map set statement is used to set the large communities in a route. It can specify a set of large communities to a route.

The *additive* keyword adds the large communities without removing the existing large communities (for standard large community-lists only).

This example shows how to set large communities.

```
route-map foo permit 10
 set large-community 1:2:3 5:6:7

route-map foo2 permit 10
 set large-community 1:2:3 5:6:7 additive
```

Matching large communities

This example shows how to match large communities.

```
route-map foo permit 10
 match large-community 1

route-map foo2 permit 10
 match large-community 1 exact-match
```

Deleting large communities

This example shows how to delete a large community.

```
route-map foo
 set large-comm-list 1 delete
```

Numbered standard large community list

This example shows how to configure a numbered large community list.

```
ip large-community-list 1 permit 1:2:3 5:6:7
ip large-community-list 1 permit 4123456789:4123456780:4123456788
```

Named standard large community list

This example shows how to configure a named standard large community list.

```
ip large-community-list standard LG_ST permit 1:2:3 5:6:7
ip large-community-list standard LG_ST permit 4123456789:4123456780:4123456788
```

Numbered expanded large community list

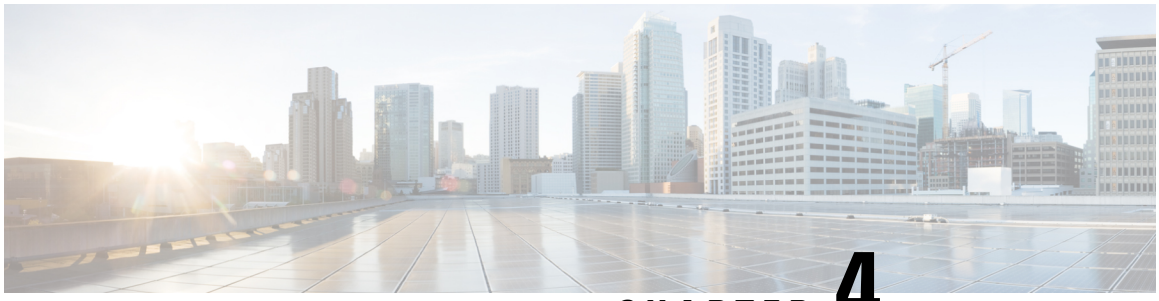
This example shows how to configure a numbered expanded large community list.

```
ip large-community-list 100 permit ^5:.*:7$
ip large-community-list 100 permit ^5:.*:8$
```

Named expanded large community list

This example shows how to configure a named expanded large community list.

```
ip large-community-list expanded LG_EX permit ^5:.*:7$
ip large-community-list expanded LG_EX permit ^5:.*:8$
```



CHAPTER 4

Configuring BGP Support for 4-byte ASN

- Feature history for BGP support for 4-byte ASN, on page 81
- Information about BGP support for 4-byte ASN, on page 81
- Configure BGP support for 4-byte ASN, on page 85
- Configuration examples for BGP support for 4-byte ASN, on page 91
- Additional references for BGP support for 4-byte ASN, on page 95

Feature history for BGP support for 4-byte ASN

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature Name and Description	Supported Platform
Cisco IOS XE 26.2.1ea	BGP Support for 4-byte ASN feature support has been introduced.	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.1	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 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.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

Information about BGP support for 4-byte ASN

The following sections provide information about BGP support for 4-byte ASN.

BGP autonomous system number formats

BGP autonomous system number formats are numbering representations that

- support four-octet autonomous system numbers allocated by IANA since January 2009
- provide two methods for displaying AS numbers: asplain and asdot notation, and
- maintain backward compatibility with traditional 2-byte AS numbers from 1 to 65535.

AS number representation methods

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 these 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.

This table shows the asdot only autonomous system number formatting:

Table 2: 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

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 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. These tables 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.

This table shows the default asplain 4-byte AS number format:

Table 3: 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

This table shows the asdot 4-byte AS number format:

Table 4: 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

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 url: http://www.IANA.org/.

Cisco implementation of 4-byte autonomous system numbers

Cisco implementation of 4-byte autonomous system numbers is a BGP feature that

- uses asplain format AS the default regular expression match and output display format for AS numbers
- supports configuration in both asplain and asdot formats AS described in RFC 5396, and
- provides gradual transition support from 2-byte AS numbers to 4-byte AS numbers according to RFC 4893.

Format configuration and display options

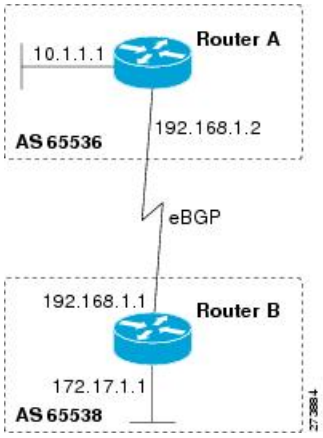
The Cisco implementation uses asplain—65538, for example—AS the default regular expression match and output display format for AS numbers, but you can configure 4-byte AS numbers in both the asplain format and the asdot format AS described in RFC 5396. To change the default regular expression match and 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.

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.

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 1: BGP peers in two autonomous systems using 4-Byte numbers



Configure BGP support for 4-byte ASN

The following sections provide configurational information about BGP support for 4-byte ASN.

Configure a BGP routing process and peers using 4-byte autonomous system numbers

This task configures a BGP routing process and BGP peers for the default IPv4 unicast address family when the BGP peers are located in an autonomous system that uses 4-byte AS numbers. The 4-byte AS numbers in this task are formatted in the default asplain (decimal value) format.

The configuration is done at Router B in the figure above (in the "Cisco Implementation of 4-Byte Autonomous System Numbers" section). 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.

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

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router BGP** *autonomous-system-number*

Example:

```
Device(config)# router bgp 65538
```

Enters router configuration mode for the specified routing process.

In this example, the 4-byte AS number, 65538, is defined in asplain notation.

Step 4 **neighbor** *IP-address* **remote-AS** *autonomous-system-number*

Example:

```
Device(config-router)# neighbor 192.168.1.2 remote-as 65536
```

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

In this example, the 4-byte AS number, 65536, is defined in asplain notation.

Step 5 Repeat Step 4 to define other BGP neighbors, AS required.

--

Step 6 **address-family ipv4** [**unicast** | **multicast** | **VRF** *VRF-name*]

Example:

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

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

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

Step 7 **neighbor** *IP-address* **activate**

Example:

```
Device(config-router-af)# neighbor 192.168.1.2 activate
```

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

Step 8 Repeat Step 7 to activate other BGP neighbors, AS required.

--

Step 9 **network** *network-number* [**mask** *network-mask*] [**route-map** *route-map-name*]

Example:

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

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

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

Step 10 **end**

Example:

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

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

Step 11 **show IP BGP** [*network*] [*network-mask*]

Example:

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

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

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

Step 12 **show IP BGP summary**

Example:

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

(Optional) Displays the status of all BGP connections.

The output from the **show IP BGP** command at Router B shows the BGP routing table entry for network 10.1.1.0 that is 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 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
```

Modify the default output and regular expression match format for 4-byte autonomous system numbers

This task modifies 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.

Before you begin

Follow these steps to modify the default output and regular expression match format for 4-byte autonomous system numbers:

Procedure

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 [**no**] **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.

Use the no form of this command to reset 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 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.

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

```
Device# 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.

```
Device# 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.

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

Device# 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

The following sections provide configuration examples for BGP support for 4-byte ASN.

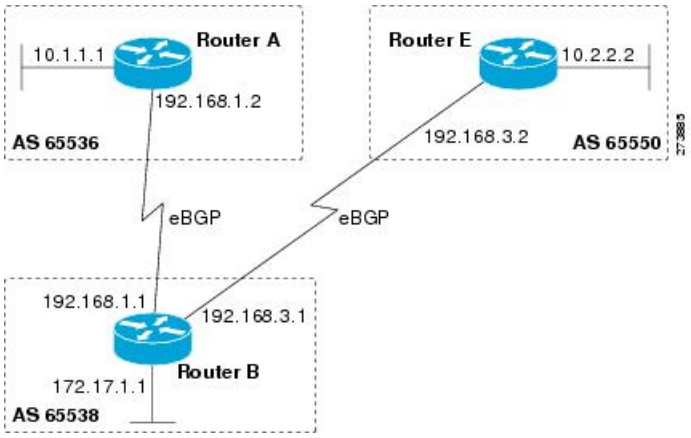
Examples: configure a BGP routing process and peers using 4-byte autonomous system numbers

These examples demonstrate how to configure Border Gateway Protocol (BGP) routing processes and peer relationships using 4-byte autonomous system numbers in both asplain and asdot notation formats. The configurations show three routers (Router A, Router B, and Router E) establishing BGP peering relationships and exchanging IPv4 unicast routes.

Asplain format

This 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 2: 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
```

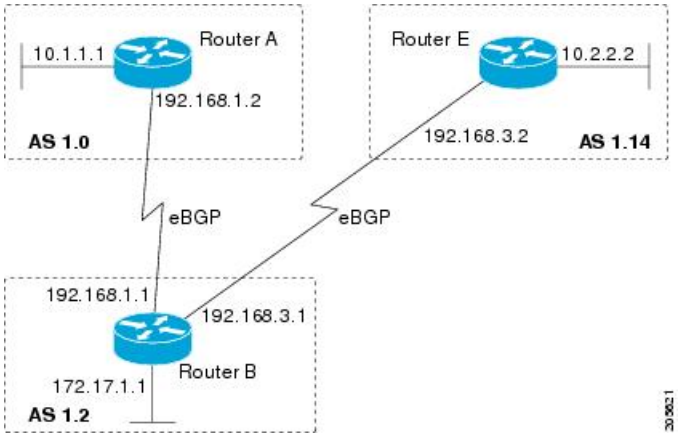
Router E

```
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

This 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 3: 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: configure a VRF and set an extended community using a BGP 4-Byte autonomous system number

This reference provides configuration examples that show how to create a VRF with route targets and route distinguishers that use 4-byte autonomous system numbers, and how to set route targets to extended community values for routes permitted by route maps.

This example 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:

```
DeviceB# 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

This 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:

```
DeviceB# 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

This example 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 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.

```
DeviceB# 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

This 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

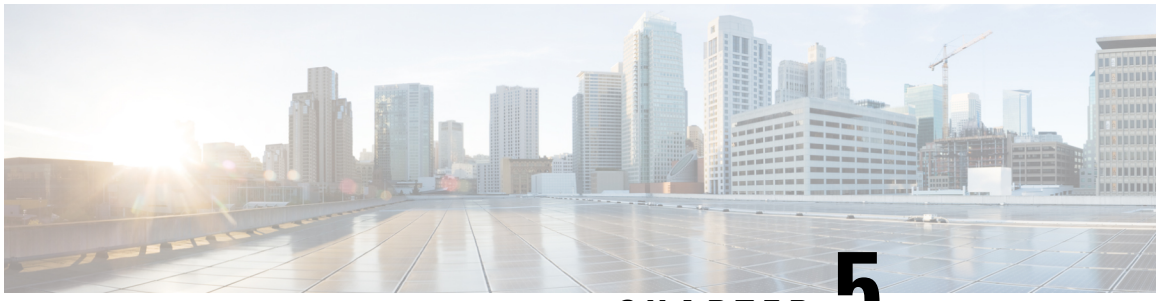
This reference provides supplementary documentation and standards information for implementing BGP support for 4-byte Autonomous System Numbers (ASN).

Related documents

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*



CHAPTER 5

Configuring BGP Monitoring Protocol

- Feature history for BGP Monitoring Protocol, on page 97
- BGP Monitoring Protocol, on page 98
- Configure a BGP Monitoring Protocol session, on page 99
- Configure BGP Monitoring Protocol servers, on page 104
- Configure BGP Monitoring Protocol on VRF neighbors, on page 107
- Verify BGP Monitoring Protocol, on page 108
- Monitor BGP Monitoring Protocol, on page 109
- Additional references for BGP Monitoring Protocol, on page 109

Feature history for BGP Monitoring Protocol

This table provides release and related information for the features explained in this module.

These features are available on all the releases subsequent to the one they were introduced in, unless noted otherwise.

Table 5: Lookup information

Release	Feature	Feature information
Cisco IOS XE 26.2.1ea	BGP Monitoring Protocol: BGP Monitoring Protocol feature support has been introduced.	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.1	The BGP Monitoring Protocol feature supports configuring devices to function as BMP servers, monitoring BGP neighbors and generating statistics reports for BGP neighbors. BMP also performs appropriate error handling, graceful scale up and closing connectivity between BMP servers and BGP neighbors.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, use https://cfnng.cisco.com/.

BGP Monitoring Protocol

The BGP Monitoring Protocol (BMP) is a monitoring protocol that:

- enables a device to function as a BMP server that monitors BGP neighbors, called BMP clients;
- allows BMP clients to connect to one or more BMP servers for monitoring active peer sessions; and
- supports multiple BMP servers that operate independently and simultaneously to monitor BMP clients.

Additional reference information

The BMP feature enables monitoring of BGP neighbors. A device can function as a BMP server that monitors one or more BMP clients. Each BMP client can have several active peer sessions configured and can connect to one or more BMP servers.

Each BMP server is specified by a number. Administrators use the command-line interface (CLI) to configure parameters such as the IP address and port number. Upon activation, a BMP server 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, send data to specific BMP servers for monitoring purposes. These clients are configured in a queue. When a connection request arrives from BMP clients to BMP servers, the connection is established based on the order in which the requests arrived.

Delay timers manage connection attempts and help avoid excessive network traffic and device load. Administrators can configure these delay timers:

- **initial-delay** delays initial connection requests.
- **failure-retry-delay** delays retry requests after a connection fails.
- **flapping-delay** delays retries after repeated connection failures.

To avoid excessive load on the device, BMP servers send route refresh requests to individual BMP clients one at a time, in the order in which connections are established in the queue. When a connected BMP client 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.

When several BMP servers activate in quick succession, administrators can use **bmp initial-refresh delay** to delay 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. Administrators can use **bmp initial-refresh skip** to skip all refresh requests from BMP servers and monitor incoming messages from the peers.

In a client-server configuration, the resource load of the devices must be kept minimal and excessive network traffic must be avoided. Administrators can configure BMP server delay timers to avoid flapping during connection between the server and client. Administrators can also configure the maximum buffer limit for the BMP session to avoid excessive message throughput or high usage of system resources.

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.

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.

Configure 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.

Before you begin

To configure a BGP monitoring protocol session, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode. Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *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.

Step 5 **end**

Example:

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

Returns to privileged EXEC mode.

Configuration examples for BGP Monitoring Protocol

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.

Examples: Configuring BGP Monitoring Protocol

This 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
```

This 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
```

This example shows 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
```

This example shows how to activate BMP on a VRF neighbor with IP address 10.1.1.1, 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)# address-family 10.1.1.1 vrf vrf1
Device(config-router)# neighbor 10.1.1.1 bmp-activate server 1 server 2
Device(config-router)# end
```

Examples: Verifying BGP Monitoring Protocol

This sample output is 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
```

This sample output is 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
```

This sample output is 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
```

This sample output is 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
```

This sample output is 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 several BGP BMP neighbors are monitored by the BMP servers, for example 10, 10 statistics messages are received by both servers in each periodic cycle that is configured.

This sample output is 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

This 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
```

This sample output is 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
```

Configure BGP Monitoring Protocol servers

Perform this task to configure BGP Monitoring Protocol (BMP) servers and its parameters in BMP server configuration mode.

Before you begin

To configure BGP monitoring protocol servers, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode. Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *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:

```
Device(config-router-bmpsrvr)# set ip dscp 5
```

Configures the IP Differentiated Services Code Point (DSCP) values for BMP servers.

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**

Example:

```
Device(config-router-bmpsrvr)# exit-bmp-server-mode
```

Exits from BMP server configuration mode and returns to router configuration mode.

Step 15 Repeat Steps 1 to 14 to configure other BMP servers in the session.

Step 16 **end**

Example:

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

Returns to privileged EXEC mode.

Configure BGP Monitoring Protocol on VRF neighbors

Perform this task to activate BGP Monitoring Protocol (BMP) on VRF neighbors.

Before you begin

To configure BGP monitoring protocol on VRF neighbors, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode. Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **router bgp** *as-number*

Example:

```
Device(config)# router bgp 65000
```

Enters router configuration mode and creates a BGP routing process.

Step 4 **address-family** { **ipv4** | **ipv6** } **vrf** *vrf-name*

Example:

```
Device (config-router)# address-family 10.1.1.1 vrf vrf1
```

Enters address family configuration mode and specifies the name of the VPN routing and forwarding (VRF) instance to associate with address family configuration mode commands.

Step 5 **neighbor** {*ipv4-addr* | *neighbor-tag* | *ipv6-addr*} **bmp-activate** {**all** | **server** *server-number-1* [**server** *server-number-2* . . . [**server** *server-number-n*]]}

Example:

```
Device(config-router)# neighbor 10.1.1.1 bmp-activate server 1 server 2
```

Activates BMP monitoring on a VRF neighbor.

Step 6 Repeat Steps 1 to 5 to configure other VRF neighbors in the session.

Step 7 **end**

Example:

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

Returns to privileged EXEC mode.

Verify BGP Monitoring Protocol

Use this task to verify the configuration for the BGP Monitoring Protocol (BMP) servers and BMP clients:

Before you begin

To verify BGP monitoring protocol, perform this procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode. Enter your password if prompted.

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.

Monitor BGP Monitoring Protocol

Use this task to enable debugging and monitor the BGP Monitoring Protocol (BMP) servers.

Before you begin

To monitor BGP monitoring protocol, perform this procedure:

Procedure

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.

Additional references for BGP Monitoring Protocol

Cisco IOS commands

Related Documents

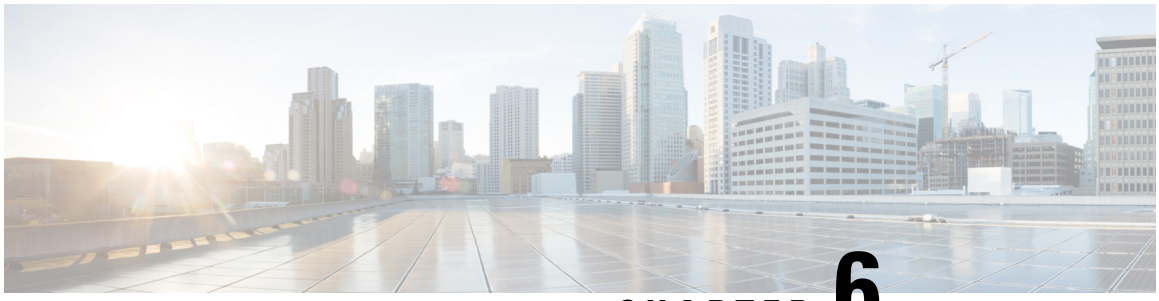
Table 6: Lookup information

Related Topic	Document Title
Cisco IOS commands	Cisco IOS command list, all releases
BGP commands	Cisco IOS IP Routing: BGP Command Reference

Technical Assistance

Table 7: Lookup information

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support



CHAPTER 6

Configuring BGP Next Hop Unchanged

- Feature history for BGP next hop unchanged, on page 111
- Information about BGP next hop unchanged, on page 111
- Restrictions for BGP next hop unchanged, on page 112
- How to configure BGP next hop unchanged, on page 112
- Configuration example for BGP next hop unchanged for an eBGP peer, on page 115

Feature history for BGP next hop unchanged

This table provides release and related information for the features explained in this module.

These features are available in all the releases subsequent to the one they were introduced in, unless noted otherwise.

Release	Feature	Feature information
Cisco IOS XE 26.2.1ea	BGP Next Hop Unchanged: BGP Next Hop Unchanged feature support has been introduced.	Cisco C9550 Series Smart Switches
Cisco IOS XE 17.18.1	The BGP Next Hop Unchanged feature allows BGP to send an update to an eBGP multihop peer with the next hop attribute unchanged.	Cisco C9350 Series Smart Switches Cisco C9610 Series Smart Switches

Use the Cisco Feature Navigator to find information about platform and software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/

Information about 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 you configure the BGP Next Hop Unchanged feature, BGP sends 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.

Use cases for BGP next hop unchanged

The BGP Next Hop Unchanged feature provides flexibility when you design and migrate networks. You can use it only between eBGP peers that are configured as multihop. You can use it in a variety of scenarios between two autonomous systems. One scenario is when multiple autonomous systems that share the same IGP are connected, 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 route reflectors.

Another common use of this feature is a VPNv4 inter-AS Option C configuration, as defined in RFC 4364, Section 10. In this configuration, VPNv4 routes are passed among autonomous systems between route reflectors of different autonomous systems. The route reflectors are several hops apart, and have **neighbor next-hop unchanged** configured. Provider edge routers of different autonomous systems establish a label-switched path (LSP) between them, through a common IGP or by advertising the next hops that lead to the provider edge routers through labeled routes among the autonomous system boundary routers. Provider edge routers are able to reach the next hops of the provider edge routers in another autonomous system through the LSPs, and can therefore install the VPNv4 routes in the VRF RIB.

Restrictions for BGP next hop unchanged

You can configure the BGP Next Hop Unchanged feature only between multihop eBGP peers. The following error message appears 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

The following procedures contain the steps to configure BGP next hop unchanged.

Configure the BGP next hop unchanged for an eBGP peer

To configure the BGP next hop unchanged for an eBGP peer, use the following procedure:

Procedure

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

a) **router bgp** *as-number*

Example:

```
Device(config)# router bgp 65535
```

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

Step 3 **address-family** { **ipv4** | **ipv6** | **l2vpn** | **nsap** | **rtfilter** | **vpnv4** | **vpnv6** }

Example:

```
Device(config-router-af)# address-family vpnv4
```

Enters address family configuration mode to configure BGP peers to accept address family specific configurations.

Step 4 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name* } **remote-as** *as-number*

Example:

```
Device(config-router-af)# neighbor 10.0.0.100 remote-as 65600
```

Adds an entry to the BGP neighbor table.

Step 5 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name* } **activate**

Example:

```
Device(config-router-af)# neighbor 10.0.0.100 activate
```

Enables the exchange of information with the peer.

Step 6 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name* } **ebgp-multihop** *ttl*

Example:

```
Device(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 7 **neighbor** { *ip-address* | *ipv6-address* | *peer-group-name* } **next-hop-unchanged**

Example:

```
Device(config-router-af)# neighbor 10.0.0.100 next-hop-unchanged
```

Configures the router to send BGP updates to the specified eBGP peer without modifying the next hop attribute.

Step 8 **end**

Example:

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

Exits address family configuration mode, and enters privileged EXEC mode.

Step 9 **show ip bgp**

Example:

```
Device# show ip bgp
```

(Optional) Displays entries in the BGP routing table.

The output indicates if the **neighbor next-hop-unchanged** command has been configured for the selected address.

Configure BGP next hop unchanged using route maps

To configure BGP next hop unchanged using route maps, use the following steps.

Procedure

Step 1 Configure an outbound route map for the eBGP neighbor.

To define the route map and apply the outbound policy for the neighbor, use the **set ip next-hop unchanged** command.

In the following configuration, the next hop for prefix 1.1.1.1 is not changed while sending to the eBGP neighbor 15.1.1.2:

Example:

```
enable
config terminal
router bgp 2
 bgp log-neighbor-changes
 neighbor 15.1.1.2 remote-as 3
 neighbor 15.1.1.2 ebgp-multihop 10
 !
 address-family ipv4
  neighbor 15.1.1.2 activate
  neighbor 15.1.1.2 route-map A out
  exit address-family
!
route-map A permit 10
 match ip address 1
 set ip next-hop unchanged
!
access-list 1 permit 1.1.1.1
end
```

Step 2 Configure next hop unchanged for both iBGP and eBGP path prefixes while sending to the eBGP neighbor.

To configure next hop unchanged for both iBGP and eBGP path prefixes while sending to the eBGP neighbor, use the **next-hop-unchanged allpaths** command.

In the following configuration, the next hop is not changed for both iBGP and eBGP path prefixes while sending to the eBGP neighbor 15.1.1.2:

Example:

```
enable
config terminal
router bgp 2
 bgp log-neighbor-changes
 neighbor 15.1.1.2 remote-as 3
 neighbor 15.1.1.2 ebgp-multihop 10
!
address-family ipv4
 neighbor 15.1.1.2 activate
 neighbor 15.1.1.2 next-hop-unchanged allpaths
 exit address-family
!
end
```

Configuration example for BGP next hop unchanged for an eBGP peer

The following example configures a multihop eBGP peer at 10.0.0.100 in a remote autonomous system. When the local router sends updates to that peer, it sends 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, and L2VPN, support the **next-hop unchanged** command. However, for the address family L2VPN BGP VPLS signaling, you must use the **next-hop self** command for proper functioning.