Configuring Advanced BGP

This chapter describes how to configure advanced features of the Border Gateway Protocol (BGP) on the Cisco NX-OS device.

This chapter includes the following sections:

- About Advanced BGP, page 10-1
- Licensing Requirements for Advanced BGP, page 10-12
- Prerequisites for Advanced BGP, page 10-12
- Guidelines and Limitations for Advanced BGP, page 10-12
- Default Settings for Advanced BGP, page 10-13
- Configuring Advanced BGP, page 10-13
- Verifying the Advanced BGP Configuration, page 10-52
- Monitoring BGP Statistics, page 10-53
- Configuration Examples, page 10-54
- Related Topics, page 10-54
- Additional References, page 10-54

About Advanced BGP

BGP is an interdomain routing protocol that provides loop-free routing between organizations or autonomous systems. Cisco NX-OS supports BGP version 4. BGP version 4 includes multiprotocol extensions that allow BGP to carry routing information for IP multicast routes and multiple Layer 3 protocol address families. BGP uses TCP as a reliable transport protocol to create TCP sessions with other BGP-enabled devices called BGP peers. When connecting to an external organization, the router creates external BGP (eBGP) peering sessions. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

This section includes the following topics:

- Peer Templates, page 10-2
- Authentication, page 10-2
- Route Policies and Resetting BGP Sessions, page 10-3
- eBGP, page 10-3
- iBGP, page 10-3

- Capabilities Negotiation, page 10-6
- Route Dampening, page 10-6
- Load Sharing and Multipath, page 10-6
- BGP Additional Paths, page 10-7
- Route Aggregation, page 10-8
- BGP Conditional Advertisement, page 10-8
- BGP Next-Hop Address Tracking, page 10-8
- Route Redistribution, page 10-9
- BFD, page 10-9
- Tuning BGP, page 10-10
- Multiprotocol BGP, page 10-10
- Graceful Restart and High Availability, page 10-11
- Low Memory Handling, page 10-11
- Virtualization Support, page 10-12

Peer Templates

BGP peer templates allow you to create blocks of common configuration that you can reuse across similar BGP peers. Each block allows you to define a set of attributes that a peer then inherits. You can choose to override some of the inherited attributes as well, making it a very flexible scheme for simplifying the repetitive nature of BGP configurations.

Cisco NX-OS implements three types of peer templates:

- The peer-session template defines BGP peer session attributes, such as the transport details, remote autonomous system number of the peer, and session timers. A peer-session template can also inherit attributes from another peer-session template (with locally defined attributes that override the attributes from an inherited peer-session).
- A peer-policy template defines the address-family dependent policy aspects for a peer including the inbound and outbound policy, filter lists, and prefix lists. A peer-policy template can inherit from a set of peer-policy templates. Cisco NX-OS evaluates these peer-policy templates in the order specified by the preference value in the inherit configuration. The lowest number is preferred over higher numbers.
- The peer template can inherit the peer-session and peer-policy templates to allow for simplified peer definitions. It is not mandatory to use a peer template but it can simplify the BGP configuration by providing reusable blocks of configuration.

Authentication

You can configure authentication for a BGP neighbor session. This authentication method adds an MD5 authentication digest to each TCP segment sent to the neighbor to protect BGP against unauthorized messages and TCP security attacks.

Note The MD5 password must be identical between BGP peers.

Route Policies and Resetting BGP Sessions

You can associate a route policy to a BGP peer. Route policies use route maps to control or modify the routes that BGP recognizes. You can configure a route policy for inbound or outbound route updates. The route policies can match on different criteria, such as a prefix or AS_path attribute, and selectively accept or deny the routes. Route policies can also modify the path attributes.

When you change a route policy applied to a BGP peer, you must reset the BGP sessions for that peer. Cisco NX-OS supports the following mechanisms to reset BGP peering sessions:

- Hard reset—A hard reset tears down the specified peering sessions, including the TCP connection, and deletes routes coming from the specified peer. This option interrupts packet flow through the BGP network. Hard reset is disabled by default.
- Soft reconfiguration inbound—A soft reconfiguration inbound triggers routing updates for the specified peer without resetting the session. You can use this option if you change an inbound route policy. Soft reconfiguration inbound saves a copy of all routes received from the peer before processing the routes through the inbound route policy. If you change the inbound route policy, Cisco NX-OS passes these stored routes through the modified inbound route policy to update the route table without tearing down existing peering sessions. Soft reconfiguration inbound can use significant memory resources to store the unfiltered BGP routes. Soft reconfiguration inbound is disabled by default.
- Route Refresh—A route refresh updates the inbound routing tables dynamically by sending route refresh requests to supporting peers when you change an inbound route policy. The remote BGP peer responds with a new copy of its routes that the local BGP speaker processes with the modified route policy. Cisco NX-OS automatically sends an outbound route refresh of prefixes to the peer.
- BGP peers advertise the route refresh capability as part of the BGP capability negotiation when establishing the BGP peer session. Route refresh is the preferred option and enabled by default.

Note BGP also uses route maps for route redistribution, route aggregation, route dampening, and other features. See Chapter 15, “Configuring Route Policy Manager,” for more information on route maps.

eBGP

External BGP (eBGP) allows you to connect BGP peers from different autonomous systems to exchange routing updates. Connecting to external networks enables traffic from your network to be forwarded to other networks and across the Internet.

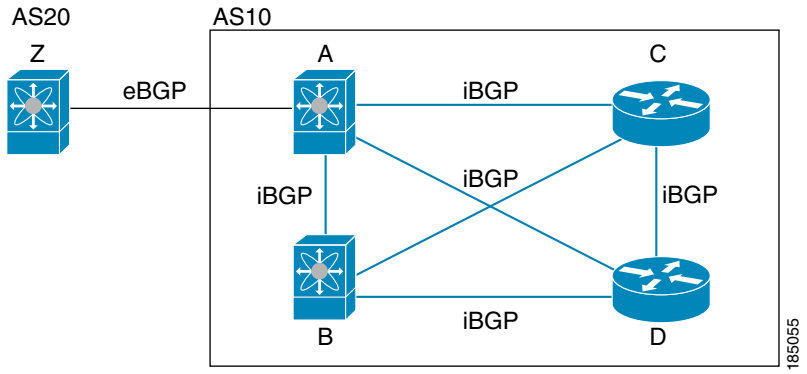
Typically eBGP peerings need to be over directly connected interfaces so that convergence will be faster when the interface goes down.

iBGP

Internal BGP (iBGP) allows you to connect BGP peers within the same autonomous system. You can use iBGP for multihomed BGP networks (networks that have more than one connection to the same external autonomous system).

Figure 10-1 shows an iBGP network within a larger BGP network.

Figure 10-1 iBGP Network



iBGP networks are fully meshed. Each iBGP peer has a direct connection to all other iBGP peers to prevent network loops.

For single-hop iBGP peers with update-source configured under neighbor configuration mode, the peer supports fast external fallover.

You should use loopback interfaces for establishing eBGP peering sessions because loopback interfaces are less susceptible to interface flapping. An interface flap occurs when the interface is administratively brought up or down because of a failure or maintenance issue. See the "Configuring eBGP" section on page 10-29 for information on multihop, fast external fallovers, and limiting the size of the AS_path attribute.

Note You should configure a separate interior gateway protocol in the iBGP network.

This section includes the following topics:

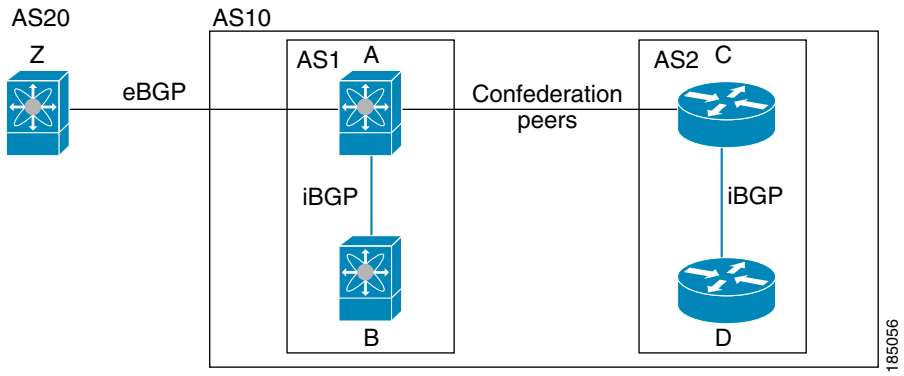
- AS Confederations, page 10-4
- Route Reflector, page 10-5

AS Confederations

A fully meshed iBGP network becomes complex as the number of iBGP peers grows. You can reduce the iBGP mesh by dividing the autonomous system into multiple subautonomous systems and grouping them into a single confederation. A confederation is a group of iBGP peers that use the same autonomous system number to communicate to external networks. Each subautonomous system is fully meshed within itself and has a few connections to other subautonomous systems in the same confederation.

Figure 10-2 shows the BGP network from Figure 10-1, split into two subautonomous systems and one confederation.

Figure 10-2 AS Confederation



In this example, AS10 is split into two subautonomous systems, AS1 and AS2. Each subautonomous system is fully meshed, but there is only one link between the subautonomous systems. By using AS confederations, you can reduce the number of links compared to the fully meshed autonomous system in Figure 10-1.

Route Reflector

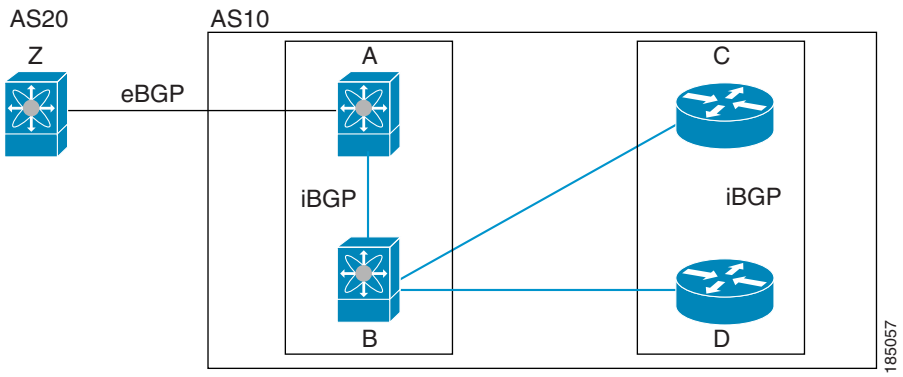
You can alternately reduce the iBGP mesh by using a route reflector configuration where route reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

Figure 10-1 shows a simple iBGP configuration with four meshed iBGP speakers (routers A, B, C, and D). Without route reflectors, when router A receives a route from an external neighbor, it advertises the route to all three iBGP neighbors.

When you configure an iBGP peer to be a route reflector, it becomes responsible for passing iBGP learned routes to a set of iBGP neighbors.

In Figure 10-3, router B is the route reflector. When the route reflector receives routes advertised from router A, it advertises (reflects) the routes to routers C and D. Router A no longer has to advertise to both routers C and D.

Figure 10-3 Route Reflector



The route reflector and its client peers form a cluster. You do not have to configure all iBGP peers to act as client peers of the route reflector. You must configure any nonclient peer as fully meshed to guarantee that complete BGP updates reach all peers.

Capabilities Negotiation

A BGP speaker can learn about BGP extensions that are supported by a peer by using the capabilities negotiation feature. Capabilities negotiation allows BGP to use only the set of features supported by both BGP peers on a link.

If a BGP peer does not support capabilities negotiation, Cisco NX-OS attempts a new session to the peer without capabilities negotiation if you have configured the address family as IPv4. Any other multiprotocol configuration (such as IPv6) requires capabilities negotiation.

Route Dampening

Route dampening is a BGP feature that minimizes the propagation of flapping routes across an internetwork. A route flaps when it alternates between the available and unavailable states in rapid succession.

For example, consider a network with three BGP autonomous systems: AS1, AS2, and AS3. Suppose that a route in AS1 flaps (it becomes unavailable). Without route dampening, AS1 sends a withdraw message to AS2. AS2 propagates the withdrawal message to AS3. When the flapping route reappears, AS1 sends an advertisement message to AS2, which sends the advertisement to AS3. If the route repeatedly becomes unavailable, and then available, AS1 sends many withdrawal and advertisement messages that propagate through the other autonomous systems.

Route dampening can minimize flapping. Suppose that the route flaps. AS2 (in which route dampening is enabled) assigns the route a penalty of 1000. AS2 continues to advertise the status of the route to neighbors. Each time that the route flaps, AS2 adds to the penalty value. When the route flaps so often that the penalty exceeds a configurable suppression limit, AS2 stops advertising the route, regardless of how many times that it flaps. The route is now dampened.

The penalty placed on the route decays until the reuse limit is reached. At that time, AS2 advertises the route again. When the reuse limit is at 50 percent, AS2 removes the dampening information for the route.

Note The router does not apply a penalty to a resetting BGP peer when route dampening is enabled, even though the peer reset withdraws the route.

Load Sharing and Multipath

BGP can install multiple equal-cost eBGP or iBGP paths into the routing table to reach the same destination prefix. Traffic to the destination prefix is then shared across all the installed paths.

The BGP best-path algorithm considers the paths as equal-cost paths if the following attributes are identical:

- Weight
- Local preference
- AS_path
- Origin code
- Multi-exit discriminator (MED)
- IGP cost to the BGP next hop

BGP supports sending and receiving multiple paths per prefix and advertising such paths. For more information, see the "BGP Additional Paths" section.

Note Paths that are received from different AS confederations are considered as equal-cost paths if the external AS_path values and the other attributes are identical.

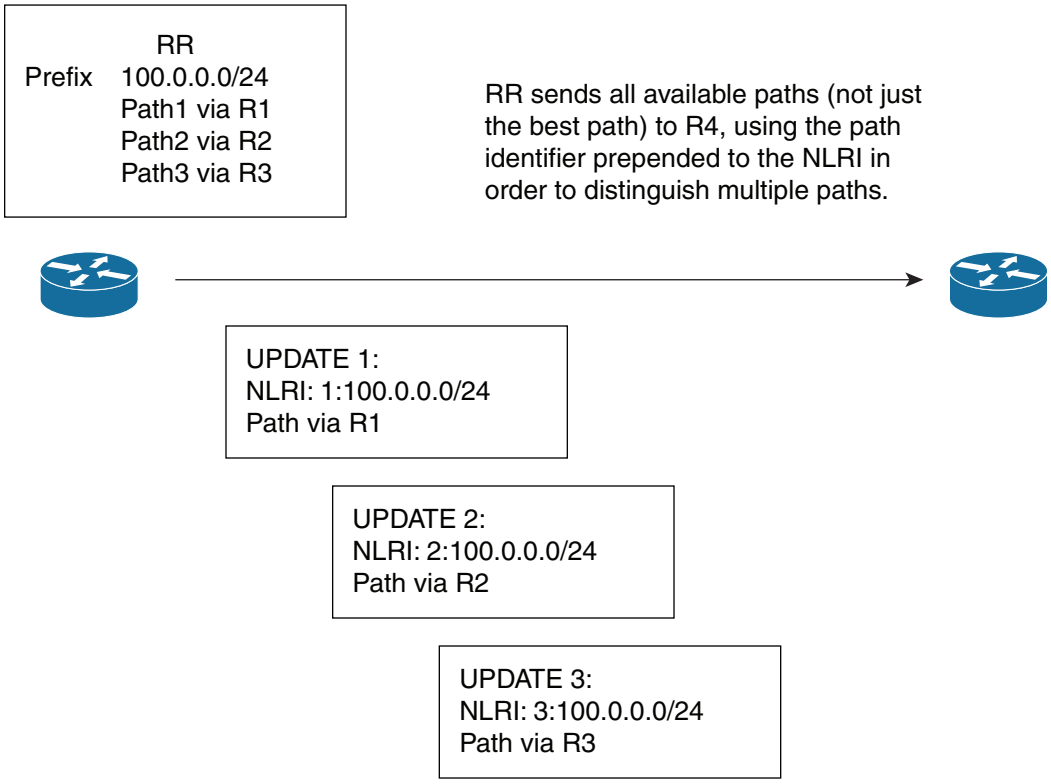
Note When you configure a route reflector for iBGP multipath, and the route reflector advertises the selected best path to its peers, the next hop for the path is not modified.

BGP Additional Paths

Only one BGP best path is advertised, and the BGP speaker accepts only one path for a given prefix from a given peer. If a BGP speaker receives multiple paths for the same prefix within the same session, it uses the most recent advertisement.

BGP supports the additional paths feature, which allows the BGP speaker to propagate and accept multiple paths for the same prefix without the new paths replacing any previous ones. This feature allows BGP speaker peers to negotiate whether they support advertising and receiving multiple paths per prefix and advertising such paths. A special 4-byte path ID is added to the network layer reachability information (NLRI) to differentiate multiple paths for the same prefix sent across a peer session. Figure 10-4 illustrates the BGP additional paths capability.

Figure 10-4 BGP Route Advertisement with the Additional Paths Capability



For information on configuring BGP additional paths, see the "Configuring BGP Additional Paths" section on page 10-26.

Route Aggregation

You can configure aggregate addresses. Route aggregation simplifies route tables by replacing a number of more specific addresses with an address that represents all the specific addresses. For example, you can replace these three more specific addresses, 10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24 with one aggregate address, 10.1.0.0/16.

Aggregate prefixes are present in the BGP route table so that fewer routes are advertised.

Note Cisco NX-OS does not support automatic route aggregation.

Route aggregation can lead to forwarding loops. To avoid this problem, when BGP generates an advertisement for an aggregate address, it automatically installs a summary discard route for that aggregate address in the local routing table. BGP sets the administrative distance of the summary discard to 220 and sets the route type to discard. BGP does not use discard routes for next-hop resolution.

BGP Conditional Advertisement

BGP conditional advertisement allows you to configure BGP to advertise or withdraw a route based on whether or not a prefix exists in the BGP table. This feature is useful, for example, in multihomed networks, in which you want BGP to advertise some prefixes to one of the providers only if information from the other provider is not present.

Consider an example network with three BGP autonomous systems: AS1, AS2, and AS3, where AS1 and AS3 connect to the Internet and to AS2. Without conditional advertisement, AS2 propagates all routes to both AS1 and AS3. With conditional advertisement, you can configure AS2 to advertise certain routes to AS3 only if routes from AS1 do not exist (if for example, the link to AS1 fails).

BGP conditional advertisement adds an exist or not-exist test to each route that matches the configured route map. See the "Configuring BGP Conditional Advertisement" section on page 10-38 for more information.

BGP Next-Hop Address Tracking

BGP monitors the next-hop address of installed routes to verify next-hop reachability and to select, install, and validate the BGP best path. BGP next-hop address tracking speeds up this next-hop reachability test by triggering the verification process when routes change in the Routing Information Base (RIB) that may affect BGP next-hop reachability.

BGP receives notifications from the RIB when the next-hop information changes (event-driven notifications). BGP is notified when any of the following events occurs:

- The next hop becomes unreachable.
- The next hop becomes reachable.
- The fully recursed Interior Gateway Protocol (IGP) metric to the next hop changes.
- The first hop IP address or first hop interface changes.

- The next hop becomes connected.
- The next hop becomes unconnected.
- The next hop becomes a local address.
- The next hop becomes a nonlocal address.

Note Reachability and recursed metric events trigger a best-path recalculation.

Event notifications from the RIB are classified as critical and noncritical. Notifications for critical and noncritical events are sent in separate batches. However, a noncritical event is sent with the critical events if the noncritical event is pending and there is a request to read the critical events.

- Critical events are related to next-hop reachability, such as the loss of next hops resulting in a switchover to a different path. A change in the IGP metric for a next hop resulting in a switchover to a different path can also be considered a critical event.
- Non-critical events are related to next hops being added without affecting the best path or changing the IGP metric to a single next hop.

See the "Configuring BGP Next-Hop Address Tracking" section on page 10-24 for more information.

Route Redistribution

You can configure BGP to redistribute static routes or routes from other protocols. You must configure a route map with the redistribution to control which routes are passed into BGP. A route map allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. See Chapter 15, "Configuring Route Policy Manager," for more information. iBGP is not redistributed to IGP by default.

You can use route maps to override the default behavior in both scenarios, but be careful when doing so as incorrect use of route maps can result in network loops. The following examples show how to use route maps to change the default behavior.

You can change the default behavior for scenario 1 by modifying the route map as follows:

```
route-map foo permit 10
   match route-type internal
router ospf 1
   redistribute bgp 100 route-map foo
```

Similarly, you can change the default behavior for scenario 2 by modifying the route map as follows:

```
route-map foo deny 10
  match route-type internal
router ospf 1
   vrf bar
     redistribute bgp 100 route-map foo
```

BFD

This feature supports bidirectional forwarding detection (BFD) for the IPv4 address families. BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

BFD for BGP is supported on eBGP peers and iBGP single-hop peers. Configure the update-source option in neighbor configuration mode for iBGP single-hop peers using BFD.

Note BFD is not supported on other iBGP peers or for multihop eBGP peers.

See the *Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide* for more information.

Tuning BGP

You can modify the default behavior of BGP through BGP timers and by adjusting the best-path algorithm.

This section includes the following topics:

- BGP Timers, page 10-10
- Tuning the Best-Path Algorithm, page 10-10

BGP Timers

BGP uses different types of timers for neighbor session and global protocol events. Each established session has a minimum of two timers for sending periodic keepalive messages and for timing out sessions when peer keepalives do not arrive within the expected time. In addition, there are other timers for handling specific features. Typically, you configure these timers in seconds. The timers include a random adjustment so that the same timers on different BGP peers trigger at different times.

Tuning the Best-Path Algorithm

You can modify the default behavior of the best-path algorithm through optional configuration parameters, including changing how the algorithm handles the multi-exit discriminator (MED) attribute and the router ID.

Multiprotocol BGP

BGP on Cisco NX-OS supports multiple address families. Multiprotocol BGP (MP-BGP) carries different sets of routes depending on the address family. For example, BGP can carry one set of routes for IPv4 unicast routing, one set of routes for IPv4 multicast routing, and one set of routes for IPv6 multicast routing.

Note Because Multicast BGP does not propagate multicast state information, you need a multicast protocol, such as Protocol Independent Multicast (PIM).

Use the router address-family and neighbor address-family configuration modes to support multiprotocol BGP configurations. MP-BGP maintains separate RIBs for each configured address family, such as a unicast RIB and a multicast RIB for BGP.

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

Graceful Restart and High Availability

Cisco NX-OS supports nonstop forwarding and graceful restart for BGP.

You can use nonstop forwarding (NSF) for BGP to forward data packets along known routes in the Forward Information Base (FIB) while the BGP routing protocol information is being restored following a failover. With NSF, BGP peers do not experience routing flaps. During a failover, the data traffic is forwarded through intelligent modules while the standby supervisor becomes active.

If a Cisco NX-OS router experiences a cold reboot, the network does not forward traffic to the router and removes the router from the network topology. In this scenario, BGP experiences a nongraceful restart and removes all routes. When Cisco NX-OS applies the startup configuration, BGP reestablishes peering sessions and relearns the routes.

A Cisco NX-OS router that has dual supervisors can experience a stateful supervisor switchover. During the switchover, BGP uses nonstop forwarding to forward traffic based on the information in the FIB, and the system is not removed from the network topology. A router whose neighbor is restarting is referred to as a "helper." After the switchover, a graceful restart operation begins. When it is in progress, both routers reestablish their neighbor relationship and exchange their BGP routes. The helper continues to forward prefixes pointing to the restarting peer, and the restarting router continues to forward traffic to peers even though those neighbor relationships are restarting. When the restarting router has all route updates from all BGP peers that are graceful restart capable, the graceful restart is complete, and BGP informs the neighbors that it is operational again.

For single-hop iBGP peers with update-source configured under neighbor configuration mode, the peer supports fast external fall-over.

With the additional BGP paths feature, if the number of paths advertised for a given prefix is the same before and after restart, the choice of path ID guarantees the final state and removal of stale paths. If fewer paths are advertised for a given prefix after a restart, stale paths can occur on the graceful restart helper peer.

Low Memory Handling

BGP reacts to low memory for the following conditions:

- Minor alert—BGP does not establish any new eBGP peers. BGP continues to establish new iBGP peers and confederate peers. Established peers remain, but reset peers are not reestablished.
- Severe alert—BGP shuts down select established eBGP peers every two minutes until the memory alert becomes minor. For each eBGP peer, BGP calculates the ratio of total number of paths received to the number of paths selected as best paths. The peers with the highest ratio are selected to be shut down to reduce memory usage. You must clear a shutdown eBGP peer before you can bring the eBGP peer back up to avoid oscillation.

Note You can exempt important eBGP peers from this selection process.

- Critical alert—BGP gracefully shuts down all the established peers. You must clear a shutdown BGP peer before you can bring the BGP peer back up.

See the "Tuning BGP" section on page 10-44 for more information on how to exempt a BGP peer from a shutdown due to a low memory condition.

Virtualization Support

You can configure one BGP instance. BGP supports virtual routing and forwarding (VRF) instances.

Licensing Requirements for Advanced BGP

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	BGP requires an Enterprise Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for Advanced BGP

Advanced BGP has the following prerequisites:

- You must enable BGP (see the "Enabling BGP" section on page 9-11).
- You should have a valid router ID configured on the system.
- You must have an AS number, either assigned by a Regional Internet Registry (RIR) or locally administered.
- You must have reachability (such as an interior gateway protocol [IGP], a static route, or a direct connection) to the peer that you are trying to make a neighbor relationship with.
- You must explicitly configure an address family under a neighbor for the BGP session establishment.

Guidelines and Limitations for Advanced BGP

Advanced BGP has the following configuration guidelines and limitations:

- The dynamic AS number prefix peer configuration overrides the individual AS number configuration inherited from a BGP template.
- If you configure a dynamic AS number for prefix peers in an AS confederation, BGP establishes sessions with only the AS numbers in the local confederation.
- BGP sessions created through a dynamic AS number prefix peer ignore any configured eBGP multihop time-to-live (TTL) value or a disabled check for directly connected peers.
- Configure a router ID for BGP to avoid automatic router ID changes and session flaps.
- Use the maximum-prefix configuration option per peer to restrict the number of routes received and system resources used.
- Configure the update source to establish a session with eBGP multihop sessions.
- Specify a BGP route map if you configure a redistribution.
- Configure the BGP router ID within a VRF.
- If you decrease the keepalive and hold timer values, the network might experience session flaps.

- When you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an additional deny statement into the route map.
- Cisco NX-OS does not support multi-hop BFD. BFD for BGP has the following limitations:
  - BFD is supported only for eBGP peers and iBGP single-hop peers.
  - To enable BFD for iBGP single-hop peers, you must configure the update-source option on the physical interface.
  - BFD is not supported for multi-hop iBGP peers and multi-hop eBGP peers.
  - BGP supports prefix-based peers, but BFD is not supported for prefix-based peers.
- The following guidelines and limitations apply to the **remove-private-as** command:
  - It applies only to eBGP peers.
  - It can be configured only in neighbor configuration mode and not in neighbor-address-family mode.
  - If the AS-path includes both private and public AS numbers, the private AS numbers are not removed.
  - If the AS-path contains the AS number of the eBGP neighbor, the private AS numbers are not removed.
  - Private AS numbers are removed only if all AS numbers in that AS-path belong to a private AS number range. Private AS numbers are not removed if a peer's AS number or a non-private AS number is found in the AS-path segment.
- If you use the **aggregate-address** command to configure aggregate addresses and the **suppress-fib-pending** command to suppress BGP routes, lossless traffic for aggregates cannot be ensured on BGP or system triggers.
- When you enable FIB suppression on the switch and route programming fails in the hardware, BGP advertises routes that are not programmed locally in the hardware.

Default Settings for Advanced BGP

Table 10-1 lists the default settings for advanced BGP parameters.

Table 10-1 Default BGP Parameters

Parameters	Default
BGP feature	Disabled
BGP additional paths	Disabled
Hold timer	180 seconds
Keep alive interval	60 seconds
Dynamic capability	Enabled

Configuring Advanced BGP

This section includes the following topics:

- Configuring BGP Session Templates, page 10-14

- Configuring BGP Peer-Policy Templates, page 10-17
- Configuring BGP Peer Templates, page 10-19
- Configuring Prefix Peering, page 10-21
- Configuring BGP Authentication, page 10-22
- Resetting a BGP Session, page 10-23
- Modifying the Next-Hop Address, page 10-23
- Configuring BGP Next-Hop Address Tracking, page 10-24
- Configuring Next-Hop Filtering, page 10-24
- Shrinking Next-Hop Groups When A Session Goes Down, page 10-25
- Disabling Capabilities Negotiation, page 10-25
- Disabling Policy Batching, page 10-26
- Configuring BGP Additional Paths, page 10-26
- Configuring eBGP, page 10-29
- Configuring AS Confederations, page 10-31
- Configuring Route Reflector, page 10-31
- Configuring Next Hops on Reflected Routes Using an Outbound Route Map, page 10-33
- Configuring Route Dampening, page 10-35
- Configuring Load Sharing and ECMP, page 10-36
- Configuring Maximum Prefixes, page 10-36
- Configuring Dynamic Capability, page 10-37
- Configuring Aggregate Addresses, page 10-37
- Suppressing BGP Routes, page 10-37
- Configuring BGP Conditional Advertisement, page 10-38
- Configuring Route Redistribution, page 10-40
- Advertising the Default Route, page 10-41
- Configuring Multiprotocol BGP, page 10-43
- Tuning BGP, page 10-44
- Configuring a Graceful Restart, page 10-48
- Configuring Virtualization, page 10-50

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Configuring BGP Session Templates

You can use BGP session templates to simplify the BGP configuration for multiple BGP peers with similar configuration needs. BGP templates allow you to reuse common configuration blocks. You configure BGP templates first and then apply these templates to BGP peers.

With BGP session templates, you can configure session attributes such as inheritance, passwords, timers, and security.

A peer-session template can inherit from one other peer-session template. You can configure the second template to inherit from a third template. The first template also inherits this third template. This indirect inheritance can continue for up to seven peer-session templates.

Any attributes configured for the neighbor take priority over any attributes inherited by that neighbor from a BGP template.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

Note When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the **default** form of the command to reset that attribute to the default state.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*
3. **template peer-session** *template-name*
4. (Optional) **password number** *password*
5. (Optional) **timers** *keepalive hold*
6. **exit**
7. **neighbor** *ip-address* **remote-as** *as-number*
8. **inherit peer-session** *template-name*
9. (Optional) **description** *text*
10. (Optional) **show bgp peer-session** *template-name*
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *autonomous-system-number*
	
	Example:
	`switch(config)# router bgp 65535`
	`switch(config-router)#`
		Enables BGP and assigns the autonomous system number to the local BGP speaker.

	Command	Purpose
Step 3	**template peer-session** *template-name* **Example**: switch(config-router)# template peer-session BaseSession switch(config-router-stmp)#	Enters peer-session template configuration mode.
Step 4	**password** *number password* **Example**: switch(config-router-stmp)# password 0 test	(Optional) Adds the cleartext password *test* to the neighbor. The password is stored and displayed in type 3 encrypted form (3DES).
Step 5	**timers** *keepalive hold* **Example**: switch(config-router-stmp)# timers 30 90	(Optional) Adds the BGP keepalive and holdtimer values to the peer-session template. The default keepalive interval is 60. The default hold time is 180.
Step 6	**exit** **Example**: switch(config-router-stmp)# exit switch(config-router)#	Exits peer-session template configuration mode.
Step 7	**neighbor** *ip-address* **remote-as** *as-number* **Example:** switch(config-router)# neighbor 192.168.1.2 remote-as 65535 switch(config-router-neighbor)#	Places the router in the neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 8	**inherit peer-session** *template-name*
	
	Example:
	switch(config-router-neighbor)# inherit peer-session BaseSession
	switch(config-router-neighbor)
		Applies a peer-session template to the peer.

Step 9	**description** *text*
	
	Example:
	switch(config-router-neighbor)# description Peer Router A
	switch(config-router-neighbor)
		(Optional) Adds a description for the neighbor.

Step 10	**show bgp peer-session** *template-name* **Example:** switch(config-router-neighbor)# show bgp peer-session BaseSession	(Optional) Displays the peer-policy template.
Step 11	**copy running-config startup-config** **Example:** switch(config-router-neighbor)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied.

This example shows how to configure a BGP peer-session template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# template peer-session BaseSession
switch(config-router-stmp)# timers 30 90
switch(config-router-stmp)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65535
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# description Peer Router A
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor)# copy running-config startup-config
```

Configuring BGP Peer-Policy Templates

You can configure a peer-policy template to define attributes for a particular address family. You assign a preference to each peer-policy template and these templates are inherited in the order specified, for up to five peer-policy templates in a neighbor address family.

Cisco NX-OS evaluates multiple peer policies for an address family using the preference value. The lowest preference value is evaluated first. Any attributes configured for the neighbor take priority over any attributes inherited by that neighbor from a BGP template.

Peer-policy templates can configure address family-specific attributes such as AS-path filter lists, prefix lists, route reflection, and soft reconfiguration.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

Note When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the default form of the command to reset that attribute to the default state.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*
3. **template peer-policy** *template-name*
4. (Optional) **advertise-active-only**
5. (Optional) **maximum-prefix** *number*
6. **exit**
7. **neighbor** *ip-address* **remote-as** *as-number*
8. **address-family** {**ipv4** | **ipv6**} {**multicast** | **unicast**}
9. **inherit peer-policy** *template-name preference*
10. (Optional) **show bgp peer-policy** *template-name*
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system-number* **Example:** switch(config)# router bgp 65535 switch(config-router)#	Enables BGP and assigns the autonomous system number to the local BGP speaker.
Step 3	**template peer-policy** *template-name* **Example:** switch(config-router)# template peer-policy BasePolicy switch(config-router-ptmp)#	Creates a peer-policy template.
Step 4	**advertise-active-only** **Example:** switch(config-router-ptmp)# advertise-active-only	(Optional) Advertises only active routes to the peer.
Step 5	**maximum-prefix** *number* **Example:** switch(config-router-ptmp)# maximum-prefix 20	(Optional) Sets the maximum number of prefixes allowed from this peer.
Step 6	**exit** **Example:** switch(config-router-ptmp)# exit switch(config-router)#	Exits peer-policy template configuration mode.
Step 7	**neighbor** *ip-address* **remote-as** *as-number* **Example:** switch(config-router)# neighbor 192.168.1.2 remote-as 65535 switch(config-router-neighbor)#	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 8	**address-family** {**ipv4** | **ipv6**} {**multicast** | **unicast**} **Example:** switch(config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	Enters global address-family configuration mode for the specified address family.
Step 9	**inherit peer-policy** *template-name preference* **Example:** switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1	Applies a peer-policy template to the peer address-family configuration and assigns the preference value for this peer policy.

	Command	Purpose
Step 10	**show bgp peer-policy** *template-name* **Example:** switch(config-router-neighbor-af)# show bgp peer-policy BasePolicy	(Optional) Displays the peer-policy template.
Step 11	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied.

This example shows how to configure a BGP peer-policy template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# template peer-session BasePolicy
switch(config-router-ptmp)# maximum-prefix 20
switch(config-router-ptmp)# exit
switch(config-router)# neighbor 192.168.1.1 remote-as 65535
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy
switch(config-router-neighbor-af)# copy running-config startup-config
```

Configuring BGP Peer Templates

You can configure BGP peer templates to combine session and policy attributes in one reusable configuration block. Peer templates can also inherit peer-session or peer-policy templates. Any attributes configured for the neighbor take priority over any attributes inherited by that neighbor from a BGP template. You configure only one peer template for a neighbor, but that peer template can inherit peer-session and peer-policy templates.

Peer templates support session and address family attributes, such as eBGP multihop time-to-live, maximum prefix, next-hop self, and timers.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

Note When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the default form of the command to reset that attribute to the default state.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*
3. **template peer** *template-name*
4. **inherit peer-session** *template-name*
5. **address-family** {**ipv4** | **ipv6**} {**multicast** | **unicast**}

6. **inherit peer** *template-name*
7. **exit**
8. **timers** *keepalive hold*
9. **exit**
10. **neighbor** *ip-address* **remote-as** *as-number*
11. **inherit peer** *template-name*
12. **timers** *keepalive hold*
13. (Optional) **show bgp peer-template** *template-name*
14. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *autonomous-system-number* **Example:** `switch(config)# router bgp 65535`	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**template peer** *template-name*
	
	Example:
	`switch(config-router)# template peer BasePeer`
	`switch(config-router-neighbor)#`
		Enters peer template configuration mode.

Step 4	**inherit peer-session** *template-name* **Example:** `switch(config-router-neighbor)# inherit peer-session BaseSession`	(Optional) Inherits a peer-session template in the peer template.

Step 5	**address-family** {**ipv4** | **ipv6**} {**multicast** | **unicast**}
	
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		(Optional) Configures the global address-family configuration mode for the specified address family.

Step 6	**inherit peer** *template-name* **Example:** `switch(config-router-neighbor-af)# inherit peer BasePolicy`	(Optional) Applies a peer template to the neighbor address-family configuration.

Step 7	**exit**
	
	Example:
	`switch(config-router-neighbor-af)# exit`
	`switch(config-router-neighbor)#`
		Exits BGP neighbor address-family configuration mode.

	Command	Purpose
Step 8	**timers** *keepalive hold* **Example**: `switch(config-router-neighbor)# timers 45 100`	(Optional) Adds the BGP timer values to the peer. These values override the timer values in the peer-session template, BaseSession.

Step 9	**exit**
	
	Example:
	`switch(config-router-neighbor)# exit`
	`switch(config-router)#`
		Exits BGP peer template configuration mode.

Step 10	**neighbor** *ip-address* **remote-as** *as-number*
	
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65535`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 11	**inherit peer** *template-name* **Example**: `switch(config-router-neighbor)# inherit peer BasePeer`	Inherits the peer template.
Step 12	**timers** *keepalive hold* **Example**: `switch(config-router-neighbor)# timers 60 120`	(Optional) Adds the BGP timer values to this neighbor. These values override the timer values in the peer template and the peer-session template.
Step 13	**show bgp peer-template** *template-name* **Example:** `switch(config-router-neighbor-af)# show bgp peer-template BasePeer`	(Optional) Displays the peer template.
Step 14	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied.

This example shows how to configure a BGP peer template and apply it to a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# template peer BasePeer
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65535
switch(config-router-neighbor)# inherit peer BasePeer
switch(config-router-neighbor)# copy running-config startup-config
```

Configuring Prefix Peering

BGP supports the definition of a set of peers using a prefix for both IPv4 and IPv6. This feature allows you to not have to add each neighbor to the configuration.

When defining a prefix peering, you must specify the remote AS number with the prefix. BGP accepts any peer that connects from that prefix and autonomous system if the prefix peering does not exceed the configured maximum peers allowed.

When a BGP peer that is part of a prefix peering disconnects, Cisco NX-OS holds its peer structures for a defined prefix peer timeout value. An established peer can reset and reconnect without danger of being blocked because other peers have consumed all slots for that prefix peering.

To configure the BGP prefix peering timeout value, use the following command in neighbor configuration mode:

Command	Purpose
timers prefix-peer-timeout *value* **Example:** `switch(config-router-neighbor)# timers prefix-peer-timeout 120`	Configures the timeout value for prefix peering. The range is from 0 to 1200 seconds. The default value is 30.

To configure the maximum number of peers, use the following command in neighbor configuration mode:

Command	Purpose
maximum-peers *value* **Example:** `switch(config-router-neighbor)# maximum-peers 120`	Configures the maximum number of peers for this prefix peering. The range is from 1 to 1000.

This example shows how to configure a prefix peering that accepts up to 10 peers:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# timers prefix-peer-timeout 120
switch(config-router)# neighbor 10.100.200.0/24 remote-as 65535
switch(config-router-neighbor)# maximum-peers 10
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

Use the **show bgp ipv4 unicast neighbors** command to show the details of the configuration for that prefix peering with a list of the currently accepted instances and the counts of active, maximum concurrent, and total accepted peers.

Configuring BGP Authentication

You can configure BGP to authenticate route updates from peers using MD5 digests.

To configure BGP to use MD5 authentication, use the following command in neighbor configuration mode:

Command	Purpose
password [**0** | **3** | **7**] *string* **Example**: switch(config-router-neighbor)# password BGPpassword	Configures an MD5 password for BGP neighbor sessions.

Resetting a BGP Session

If you modify a route policy for BGP, you must reset the associated BGP peer sessions. If the BGP peers do not support route refresh, you can configure a soft reconfiguration for inbound policy changes. Cisco NX-OS automatically attempts a soft reset for the session.

To configure soft reconfiguration inbound, use the following command in neighbor address-family configuration mode:

Command	Purpose
soft-reconfiguration inbound **Example**: switch(config-router-neighbor-af)# soft-reconfiguration inbound	Enables soft reconfiguration to store the inbound BGP route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

To reset a BGP neighbor session, use the following command in any mode:

Command	Purpose
clear bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} *ip-address* **soft** {**in** | **out**} **Example**: switch# clear bgp ip unicast 192.0.2.1 soft in	Resets the BGP session without tearing down the TCP session.

Modifying the Next-Hop Address

You can modify the next-hop address used in a route advertisement in the following ways:

- Disable next-hop calculation and use the local BGP speaker address as the next-hop address.
- Set the next-hop address as a third-party address. Use this feature in situations where the original next-hop address is on the same subnet as the peer that the route is being sent to. Using this feature saves an extra hop during forwarding.

To modify the next-hop address, use the following commands in address-family configuration mode:

Command	Purpose
next-hop-self **Example:** `switch(config-router-neighbor-af)# next-hop-self`	Uses the local BGP speaker address as the next-hop address in route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
next-hop-third-party **Example:** `switch(config-router-neighbor-af)# next-hop-third-party`	Sets the next-hop address as a third-party address. Use this command for single-hop EBGP peers that do not have **next-hop-self** configured.

Configuring BGP Next-Hop Address Tracking

BGP next-hop address tracking is enabled by default and cannot be disabled.

You can modify the delay interval between RIB checks to increase the performance of BGP next-hop tracking.

To modify the BGP next-hop address tracking, use the following commands in address-family configuration mode:

Command	Purpose
nexthop trigger-delay {**critical** | **non-critical**} *milliseconds* **Example:** `switch(config-router-af)# nexthop trigger-delay critical 5000`	Specifies the next-hop address tracking delay timer for critical next-hop reachability routes and for noncritical routes. The range is from 1 to 4294967295 milliseconds. The critical timer default is 3000. The noncritical timer default is 10000.

Configuring Next-Hop Filtering

BGP next-hop filtering allows you to specify that when a next-hop address is checked with the RIB, the underlying route for that next-hop address is passed through the route map. If the route map rejects the route, the next-hop address is treated as unreachable.

BGP marks all next hops that are rejected by the route policy as invalid and does not calculate the best path for the routes that use the invalid next-hop address.

To configure BGP next-hop filtering, use the following command in address-family configuration mode:

Command	Purpose
nexthop route-map *name* **Example:** `switch(config-router-af)# nexthop route-map nextHopLimits`	Specifies a route map to match the BGP next-hop route to. The name can be any case-sensitive, alphanumeric string up to 63 characters.

Shrinking Next-Hop Groups When A Session Goes Down

You can configure BGP to shrink ECMP groups in an accelerated way when a session goes down.

This feature applies to the following BGP path failure events:

- Any single or multiple Layer 3 link failures
- Line card failures
- BFD failure detections for BGP neighbors

Note BFDv6 is currently not supported.

- Administrative shutdown of BGP neighbors (using the **shutdown** command)

The accelerated handling of the first two events (Layer 3 link failures and line card failures) is enabled by default and does not require a configuration command to be enabled.

To configure the accelerated handling of the last two events, use the following command in router configuration mode:

Command	Purpose
neighbor-down fib-accelerate **Example**: `switch(config-router)# neighbor-down fib-accelerate`	Withdraws the corresponding next hop from all next-hop groups (ECMP groups and single next-hop routes) whenever a BGP session goes down. **Note** This command applies to both IPv4 and IPv6 address-family routes. **Note** This command is supported only in a BGP-only environment where all non-direct routes are installed by BGP.

Disabling Capabilities Negotiation

You can disable capabilities negotiations to interoperate with older BGP peers that do not support capabilities negotiation.

To disable capabilities negotiation, use the following command in neighbor configuration mode:

Command	Purpose
dont-capability-negotiate **Example**: `switch(config-router-neighbor)# dont-capability-negotiate`	Disables capabilities negotiation. You must manually reset the BGP sessions after configuring this command.

Disabling Policy Batching

In BGP deployments where prefixes have unique attributes, BGP tries to identify routes with similar attributes to bundle in the same BGP update message. To avoid the overhead of this additional BGP processing, you can disable batching.

Cisco recommends that you disable policy batching for BGP deployments that have a large number of routes with unique next hops.

To disable policy batching, use the following command in router configuration mode:

Command	Purpose
disable-policy-batching **Example:** `switch(config-router)# disable-policy-batching`	Disables the batching evaluation of prefix advertisements to all peers.

Configuring BGP Additional Paths

BGP supports sending and receiving multiple paths per prefix and advertising such paths. This section includes the following topics:

- Advertising the Capability of Sending and Receiving Additional Paths, page 10-26
- Configuring the Sending and Receiving of Additional Paths, page 10-27
- Configuring Advertised Paths, page 10-28
- Configuring Additional Path Selection, page 10-28

Advertising the Capability of Sending and Receiving Additional Paths

You can configure BGP to advertise the capability of sending and receiving additional paths to and from the BGP peers. To do so, use the following commands in neighbor address-family configuration mode:

Command	Purpose
[**no**] **capability additional-paths send** [**disable**] **Example:** `switch(config-router-neighbor-af)# capability additional-paths send`	Advertises the capability to send additional paths to the BGP peer. The **disable** option disables the advertising capability of sending additional paths. The **no** form of this command disables the capability of sending additional paths.

Command	Purpose
[**no**] **capability additional-paths receive** [**disable**] **Example**: `switch(config-router-neighbor-af)# capability additional-paths receive`	Advertises the capability to receive additional paths from the BGP peer. The **disable** option disables the advertising capability of receiving additional paths. The **no** form of this command disables the capability of receiving additional paths.
show bgp neighbor **Example**: `switch(config-router-neighbor-af)# show bgp neighbor`	Displays whether the local peer has advertised the additional paths send or receive capability to the remote peer.

This example shows how to configure BGP to advertise the capability to send and receive additional paths to the BGP peer:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# neighbor 10.131.31.2 remote-as 100
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# capability additional-paths send
switch(config-router-neighbor-af)# capability additional-paths receive
```

Configuring the Sending and Receiving of Additional Paths

You can configure the capability of sending and receiving additional paths to and from the BGP peers. To do so, use the following commands in address-family configuration mode:

Command	Purpose
[**no**] **additional-paths send** **Example**: `switch(config-router-af)# additional-paths send`	Enables the send capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled. The **no** form of this command disables the send capability.
[**no**] **additional-paths receive** **Example**: `switch(config-router-af)# additional-paths receive`	Enables the receive capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled. The **no** form of this command disables the receive capability.
show bgp neighbor **Example**: `switch(config-router-af)# show bgp neighbor`	Displays whether the local peer has advertised the additional paths send or receive capability to the remote peer.

This example shows how to enable the additional paths send and receive capability for all neighbors under the specified address family for which this capability has not been disabled:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# address-family ipv4 unicast
```

```
switch(config-router-af)# additional-paths send
switch(config-router-af)# additional-paths receive
```

Configuring Advertised Paths

You can specify the paths that are advertised for BGP. To do so, use the following commands in route-map configuration mode:

Command	Purpose
[no] set ip next-hop unchanged **Example:** `switch(config-route-map)# set ip next-hop unchanged`	Specifies an unchanged next-hop IP address.
[no] set path-selection all advertise **Example:** `switch(config-route-map)# set path-selection all advertise`	Specifies that all paths be advertised for a given prefix. The **no** form of this command specifies that only the best path be advertised.
show bgp {**ipv4** | **ipv6**} **unicast** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*] **Example:** `switch(config-route-map)# show bgp ipv4 unicast`	Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

This example shows how to specify that all paths be advertised for the specified prefix:

```
switch# configure terminal
switch(config)# route-map PATH_SELECTION_RMAP
switch(config-route-map)# match ip address prefix-list p1
switch(config-route-map)# set path-selection all advertise
```

Configuring Additional Path Selection

You can configure the capability of selecting additional paths for a prefix. To do so, use the following commands in address-family configuration mode:

Command	Purpose
[no] additional-paths selection route-map *map-name* **Example:** `switch(config-router-af)# additional-paths selection route-map map1`	Configures the capability of selecting additional paths for a prefix. The **no** form of this command disables the additional paths selection capability.
show bgp {**ipv4** | **ipv6**} **unicast** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*] **Example:** `switch(config-route-af)# show bgp ipv4 unicast`	Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

This example shows how to configure the additional paths selection under the specified address family:

```
switch# configure terminal
switch(config)# router bgp 100
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# additional-paths selection route-map PATH_SELECTION_RMAP
```

Configuring eBGP

This section includes the following topics:

- Disabling eBGP Single-Hop Checking, page 10-29
- Configuring eBGP Multihop, page 10-29
- Disabling a Fast External Fallover, page 10-30
- Limiting the AS-path Attribute, page 10-30
- Configuring Local AS Support, page 10-30

Disabling eBGP Single-Hop Checking

You can configure eBGP to disable checking whether a single-hop eBGP peer is directly connected to the local router. Use this option for configuring a single-hop loopback eBGP session between directly connected switches.

To disable checking whether or not a single-hop eBGP peer is directly connected, use the following command in neighbor configuration mode:

Command	Purpose
disable-connected-check **Example**: `switch(config-router-neighbor)# disable-connected-check`	Disables checking whether or not a single-hop eBGP peer is directly connected. You must manually reset the BGP sessions after using this command.

Configuring eBGP Multihop

You can configure the eBGP time-to-live (TTL) value to support eBGP multihop. In some situations, an eBGP peer is not directly connected to another eBGP peer and requires multiple hops to reach the remote eBGP peer. You can configure the eBGP TTL value for a neighbor session to allow these multihop sessions.

To configure eBGP multihop, use the following command in neighbor configuration mode:

Command	Purpose
ebgp-multihop *ttl-value* **Example**: `switch(config-router-neighbor)# ebgp-multihop 5`	Configures the eBGP TTL value for eBGP multihop. The range is from 2 to 255. You must manually reset the BGP sessions after using this command.

Disabling a Fast External Fallover

By default, the Cisco NX-OS device supports fast external fallover for neighbors in all VRFs and address-families (IPv4 or IPv6). Typically, when a BGP router loses connectivity to a directly connected eBGP peer, BGP triggers a fast external fallover by resetting the eBGP session to the peer. You can disable this fast external fallover to limit the instability caused by link flaps.

To disable fast external fallover, use the following command in router configuration mode:

Command	Purpose
no fast-external-fallover **Example**: `switch(config-router)# no fast-external-fallover`	Disables a fast external fallover for eBGP peers. This command is enabled by default.

Limiting the AS-path Attribute

You can configure eBGP to discard routes that have excessive AS numbers in the AS-path attribute.

To discard routes that have excessive AS numbers in the AS-path attribute, use the following command in router configuration mode:

Command	Purpose
maxas-limit *number* **Example**: `switch(config-router)# maxas-limit 50`	Discards eBGP routes that have a number of AS-path segments that exceed the specified limit. The range is from 1 to 2000.

Configuring Local AS Support

The local AS feature allows a router to appear to be a member of a second autonomous system (AS), in addition to its real AS. Local AS allows two ISPs to merge without modifying peering arrangements. Routers in the merged ISP become members of the new autonomous system but continue to use their old AS numbers for their customers.

This feature can only be used for true eBGP peers. You cannot use this feature for two peers that are members of different confederation subautonomous systems.

To configure eBGP local AS support, use the following command in neighbor configuration mode:

Command	Purpose
local-as *number* [**no-prepend** [**replace-as** [**dual-as**]]] **Example**: `switch(config-router-neighbor)# local-as 1.1`	Configures eBGP to prepend the local AS *number* to the AS_PATH attribute.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 in xx.xx format.

This example shows how to configure local AS support on a VRF:

```
switch# configure terminal
switch(config)# router bgp 1
```

```
switch(config-router)# vrf test
switch(config-router-vrf)# local-as 1
switch(config-router-vrf)# show running-config bgp
```

Configuring AS Confederations

To configure an AS confederation, you must specify a confederation identifier. To the outside world, the group of autonomous systems within the AS confederation look like a single autonomous system with the confederation identifier as the autonomous system number.

To configure a BGP confederation identifier, use the following command in router configuration mode:

Command	Purpose
confederation identifier *as-number* **Example**: `switch(config-router)# confederation identifier 4000`	Configures a confederation identifier for an AS confederation. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

To configure the autonomous systems that belong to the AS confederation, use the following command in router configuration mode:

Command	Purpose
bgp confederation peers *as-number* [*as-number2...*] **Example**: `switch(config-router)# bgp confederation peers 5 33 44`	Specifies a list of autonomous systems that belong to the confederation. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Configuring Route Reflector

You can configure iBGP peers as route reflector clients to the local BGP speaker, which acts as the route reflector. Together, a route reflector and its clients form a cluster. A cluster of clients usually has a single route reflector. In such instances, the cluster is identified by the router ID of the route reflector. To increase redundancy and avoid a single point of failure in the network, you can configure a cluster with more than one route reflector. You must configure all route reflectors in the cluster with the same 4-byte cluster ID so that a route reflector can recognize updates from route reflectors in the same cluster.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **cluster-id** *cluster-id*
4. **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}

5. (Optional) **client-to-client reflection**
6. **exit**
7. **neighbor** *ip-address* **remote-as** *as-number*
8. **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
9. **route-reflector-client**
10. (Optional) **show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors**
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number*
	
	Example:
	`switch(config)# router bgp 65535`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**cluster-id** *cluster-id* **Example:** `switch(config-router)# cluster-id 192.0.2.1`	Configures the local router as one of the route reflectors that serve the cluster. You specify a cluster ID to identify the cluster. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

Step 4	**address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
	
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters router address-family configuration mode for the specified address family.

Step 5	**client-to-client reflection** **Example:** `switch(config-router-af)# client-to-client reflection`	(Optional) Configures client-to-client route reflection. This feature is enabled by default. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

Step 6	**exit**
	
	Example:
	`switch(config-router-neighbor)# exit`
	`switch(config-router)#`
		Exits router address configuration mode.

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	
	Example:
	`switch(config-router)# neighbor 192.0.2.10 remote-as 65535`
	`switch(config-router-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

	Command	Purpose
Step 8	**address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **Example:** switch(config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	Enters neighbor address-family configuration mode for the specified address family.
Step 9	**route-reflector-client** **Example**: switch(config-router-neighbor-af)# route-reflector-client	Configures the device as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 10	**show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors** **Example:** switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors	(Optional) Displays the BGP peers.
Step 11	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure the router as a route reflector and add one neighbor as a client:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# neighbor 192.0.2.10 remote-as 65535
switch(config-router-neighbor)# address-family ip unicast
switch(config-router-neighbor-af)# route-reflector-client
switch(config-router-neighbor-af)# copy running-config startup-config
```

Configuring Next Hops on Reflected Routes Using an Outbound Route Map

You can change the next hop on reflected routes on a BGP route reflector using an outbound route map. You can configure the outbound route map to specify the peer's local address as the next-hop address.

Note The **next-hop-self** command does not enable this functionality for routes being reflected to clients by a route reflector. This functionality can only be enabled using an outbound route map.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

You must enter the **set next-hop** command to configure an address family specific next-hop address. For example, for the IPv6 address family, you must enter the **set ipv6 next-hop peer-address** command.

- When setting IPv4 next hops using route maps—If **set ip next-hop peer-address** matches the route map, the next hop is set to the peer's local address. If no next hop is set in the route map, the next hop is set to the one stored in the path.
- When setting IPv6 next hops using route maps—If **set ipv6 next-hop peer-address** matches the route map, the next hop is set as follows:

- For IPv6 peers, the next hop is set to the peer's local IPv6 address.
- For IPv4 peers, if **update-source** is configured, the next hop is set to the source interface's IPv6 address, if any. If no IPv6 address is configured, no next hop is set.
- For IPv4 peers, if **update-source** is not configured, the next hop is set to the outgoing interface's IPv6 address, if any. If no IPv6 address is configured, no next hop is set.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** *ip-address* **remote-as** *as-number*
4. (Optional) **update-source** *interface number*
5. **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
6. **route-reflector-client**
7. **route-map** *map-name* **out**
8. (Optional) **show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number*
	
	Example:
	`switch(config)# router bgp 200`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**neighbor** *ip-address* **remote-as** *as-number*
	
	Example:
	`switch(config-router)# neighbor 192.0.2.12 remote-as 200`
	`switch(config-router-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

Step 4	**update-source** *interface number* **Example:** `switch(config-router-neighbor)# update-source loopback 300`	(Optional) Specifies and updates the source of the BGP session.

Step 5	**address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
	
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters router address-family configuration mode for the specified address family.

	Command	Purpose
Step 6	**route-reflector-client** **Example**: switch(config-router-neighbor-af)# route-reflector-client	Configures the device as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 7	**route-map** *map-name* **out** **Example**: switch(config-router-neighbor-af)# route-map setrrnh out	Applies the configured BGP policy to outgoing routes.
Step 8	**show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} [**ip-address** | **ipv6-prefix**] **route-map** *map-name* [**vrf** *vrf-name*] **Example:** switch(config-router-neighbor-af)# show bgp ipv4 unicast route-map setrrnh	(Optional) Displays the BGP routes that match the route map.
Step 9	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure the next hop on reflected routes on a BGP route reflector using an outbound route map:

```
switch# configure terminal
switch(config)# interface loopback 300
switch(config-if)# ip address 192.0.2.11/32
switch(config-if)# ipv6 address 2001::a0c:1a65/64
switch(config-if)# ip router ospf 1 area 0.0.0.0
switch(config-if)# exit
switch(config)# route-map setrrnh permit 10
switch(config-route-map)# set ip next-hop peer-address
switch(config-route-map)# exit
switch(config)# route-map setrrnhv6 permit 10
switch(config-route-map)# set ipv6 next-hop peer-address
switch(config-route-map)# exit
switch(config)# router bgp 200
switch(config-router)# neighbor 192.0.2.12 remote-as 200
switch(config-router-neighbor)# update-source loopback 300
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# route-reflector-client
switch(config-router-neighbor-af)# route-map setrrnh out
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# address-family ipv6 unicast
switch(config-router-neighbor-af)# route-reflector-client
switch(config-router-neighbor-af)# route-map setrrnhv6 out
```

Configuring Route Dampening

You can configure route dampening to minimize route flaps propagating through your iBGP network.

To configure route dampening, use the following command in address-family or VRF address-family configuration mode:

Command	Purpose
dampening [{*half-life reuse-limit suppress-limit max-suppress-time* | **route-map** *map-name*}] **Example:** `switch(config-router-af)# dampening route-map bgpDamp`	Disables capabilities negotiation. The parameter values are as follows: • half-life—The range is from 1 to 45. • reuse-limit—The range is from 1 to 20000. • suppress-limit—The range is from 1 to 20000. • max-suppress-time—The range is from 1 to 255.

Configuring Load Sharing and ECMP

You can configure the maximum number of paths that BGP adds to the route table for equal-cost multipath load balancing.

To configure the maximum number of paths, use the following command in router address-family configuration mode:

Command	Purpose
maximum-paths [**ibgp**] *maxpaths* **Example:** `switch(config-router-af)# maximum-paths 8`	Configures the maximum number of equal-cost paths for load sharing. The default is 1.

Configuring Maximum Prefixes

You can configure the maximum number of prefixes that BGP can receive from a BGP peer. If the number of prefixes exceeds this value, you can optionally configure BGP to generate a warning message or tear down the BGP session to the peer.

To configure the maximum allowed prefixes for a BGP peer, use the following command in neighbor address-family configuration mode:

Command	Purpose
maximum-prefix *maximum* [*threshold*] [**restart** *time* | **warning-only**] **Example:** `switch(config-router-neighbor-af)# maximum-prefix 12`	Configures the maximum number of prefixes from a peer. The parameter ranges are as follows: • *maximum*—The range is from 1 to 300000. • *Threshold*—The range is from 1 to 100 percent. The default is 75 percent. • *time*—The range is from 1 to 65535 minutes. This command triggers an automatic notification and session reset for the BGP neighbor sessions if the prefix limit is exceeded.

Configuring Dynamic Capability

You can configure dynamic capability for a BGP peer.

To configure dynamic capability, use the following command in neighbor configuration mode:

Command	Purpose
dynamic-capability **Example**: `switch(config-router-neighbor)# dynamic-capability`	Enables dynamic capability. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Configuring Aggregate Addresses

You can configure aggregate address entries in the BGP route table.

To configure an aggregate address, use the following command in router address-family configuration mode:

Command	Purpose
aggregate-address *ip-prefix/length* [**as-set**] [**summary-only**] [**advertise-map** *map-name*] [**attribute-map** *map-name*] [**suppress-map** *map-name*] **Example**: `switch(config-router-af)# aggregate-address 192.0.2.0/8 as-set`	Creates an aggregate address. The path advertised for this route is an autonomous system set that consists of all elements contained in all paths that are being summarized: • The **as-set** keyword generates autonomous system set path information and community information from contributing paths. • The **summary-only** keyword filters all more-specific routes from updates. • The **advertise-map** keyword and argument specify the route map used to select attribute information from selected routes. • The **attribute-map** keyword and argument specify the route map used to select attribute information from the aggregate. • The **suppress-map** keyword and argument conditionally filter more specific routes.

Suppressing BGP Routes

You can configure Cisco NX-OS to advertise newly learned BGP routes only after these routes are confirmed by the Forwarding Information Base (FIB) and programmed in the hardware. After the routes are programmed, subsequent changes to these routes do not require this hardware-programming check.

To suppress BGP routes, use the following command in router configuration mode:

Command	Purpose

suppress-fib-pending
 	
Example:
`switch(config-router)#`
`suppress-fib-pending`
	Suppresses newly learned BGP routes (IPv4 or IPv6) from being advertised to downstream BGP neighbors until the routes have been programmed in the hardware.

Configuring BGP Conditional Advertisement

You can configure BGP conditional advertisement to limit the routes that BGP propagates. You define the following two route maps:

- Advertise map—Specifies the conditions that the route must match before BGP considers the conditional advertisement. This route map can contain any appropriate match statements.
- Exist map or nonexist map—Defines the prefix that must exist in the BGP table before BGP propagates a route that matches the advertise map. The nonexist map defines the prefix that must not exist in the BGP table before BGP propagates a route that matches the advertise map. BGP processes only the permit statements in the prefix list match statements in these route maps.

If the route does not pass the condition, BGP withdraws the route if it exists in the BGP table.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** *ip-address* **remote-as** *as-number*
4. **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
5. **advertise-map** *adv-map* {**exist-map** *exist-rmap* | **non-exist-map** *nonexist-rmap*}
6. (Optional) **show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors**
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number*
	
	Example:
	`switch(config)# router bgp 65535`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**neighbor** *ip-address* **remote-as** *as-number*
	
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65534`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

Step 4	**address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
	
	Example:
	`switch(config-router-neighbor)# address-family ipv4 multicast`
	`switch(config-router-neighbor-af)#`
		Enters address-family configuration mode.

Step 5	**advertise-map** *adv-map* {**exist-map** *exist-rmap* | **non-exist-map** *nonexist-rmap*} **Example**: `switch(config-router-neighbor-af)# advertise-map advertise exist-map exist`	Configures BGP to conditionally advertise routes based on the two configured route maps: • *adv-map*—Specifies a route map with **match** statements that the route must pass before BGP passes the route to the next route map. The *adv-map* is a case-sensitive, alphanumeric string up to 63 characters. • *exist-rmap*—Specifies a route map with match statements for a prefix list. A prefix in the BGP table must match a prefix in the prefix list before BGP advertises the route. The *exist-rmap* is a case-sensitive, alphanumeric string up to 63 characters. • *nonexist-rmap*—Specifies a route map with match statements for a prefix list. A prefix in the BGP table must not match a prefix in the prefix list before BGP advertises the route. The *nonexist-rmap* is a case-sensitive, alphanumeric string up to 63 characters.

	Command	Purpose
Step 6	**show bgp** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **neighbors** **Example:** switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors	(Optional) Displays information about BGP and the configured conditional advertisement route maps.
Step 7	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure BGP conditional advertisement:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# neighbor 192.0.2.2 remote-as 65534
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# advertise-map advertise exist-map exist
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# exit
switch(config-router)# exit
switch(config)# route-map advertise
switch(config-route-map)# match as-path pathList
switch(config-route-map)# exit
switch(config)# route-map exit
switch(config-route-map)# match ip address prefix-list plist
switch(config-route-map)# exit
switch(config)# ip prefix-list plist permit 172.16.201.0/27
```

Configuring Route Redistribution

You can configure BGP to accept routing information from another routing protocol and redistribute that information through the BGP network. Optionally, you can assign a default route for redistributed routes.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
4. **redistribute** {**direct** | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
5. (Optional) **default-metric** *value*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**router bgp** *as-number* Example: switch(config)# router bgp 65535 switch(config-router)#	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	**address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **Example**: switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Enters address-family configuration mode.
Step 4	**redistribute** {**direct** | {**eigrp** | **isis** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example**: switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap	Redistributes routes from other protocols into BGP. See the "Configuring Route Maps" section on page 15-11 for more information about route maps.
Step 5	**default-metric** *value* **Example**: switch(config-router-af)# default-metric 33	(Optional) Generates a default route into BGP.
Step 6	**copy running-config startup-config** **Example:** switch(config-router-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to redistribute EIGRP into BGP:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap
switch(config-router-af)# copy running-config startup-config
```

Advertising the Default Route

You can configure BGP to advertise the default route (network 0.0.0.0).

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

SUMMARY STEPS

1. **configure terminal**
2. **route-map allow permit**
3. **exit**
4. **ip route** *ip-address network-mask* **null** *null-interface-number*
5. **router bgp** *as-number*
6. **address-family** {**ipv4** | **ipv6**} **unicast**
7. **default-information originate**
8. **redistribute static route-map allow**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**route-map allow permit** Example: switch(config)# route-map allow permit switch(config-route-map)#	Enters router map configuration mode and defines the conditions for redistributing routes.
Step 3	**exit** Example: switch(config-route-map)# exit switch(config)#	Exits router map configuration mode.
Step 4	**ip route** *ip-address network-mask* **null** *null-interface-number* **Example**: switch(config)# ip route ip route 0.0.0.0/0 null 0	Configures the IP address.
Step 5	**router bgp** *as-number* **Example**: switch(config)# router bgp 65535 switch(config-router)#	Enters BGP mode and assigns the AS number to the local BGP speaker.
Step 6	**address-family** {**ipv4**	**ipv6**} **unicast** **Example**: switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Enters address family configuration mode.
Step 7	**default-information originate** **Example**: switch(config-router-af)# default-information originate	Advertises the default route.

	Command	Purpose
Step 8	**redistribute static route-map allow** **Example**: `switch(config-router-af)# redistribute static route-map allow`	Redistributes the default route.
Step 9	**copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Multiprotocol BGP

You can configure MP-BGP to support multiple address families, including IPv4 and IPv6 unicast and multicast routes.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **neighbor** *ip-address* **remote-as** *as-numbe*r
4. **address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**}
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number*
	
	Example:
	`switch(config)# router bgp 65535`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**neighbor** *ip-address* **remote-as** *as-number*
	
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65534`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

	Command	Purpose
Step 4	**address-family** {**ipv4** | **ipv6**} {**unicast** | **multicast**} **Example**: switch(config-router-neighbor)# address-family ipv4 multicast switch(config-router-neighbor-af)#	Enters address family configuration mode.
Step 5	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to enable advertising and receiving IPv4 and IPv6 routes for multicast RPF for a neighbor:

```
switch# configure terminal
switch(config)# interface ethernet 2/1
switch(config-if)# ipv6 address 2001:0DB8::1
switch(config-if)# router bgp 65535
switch(config-router)# neighbor 192.168.1.2 remote-as 35537
switch(config-router-neighbor)# address-family ipv4 multicast
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# address-family ipv6 multicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

Tuning BGP

You can tune BGP characteristics through a series of optional parameters.

To tune BGP, use the following optional commands in router configuration mode:

Command	Purpose
bestpath [**always-compare-med** | **as-path multipath-relax** | **compare-routerid** | **cost-community ignore** | **med** {**confed** | **missing-as-worst** | **non-deterministic**}] **Example**: `switch(config-router)# bestpath always-compare-med`	Modifies the best-path algorithm. The optional parameters are as follows: • **always-compare-med**—Compares MED on paths from different autonomous systems. • **as-path multipath-relax**—Allows load sharing across the providers with different (but equal-length) AS paths. Without this option, the AS paths must be identical for load sharing. • **compare-routerid**—Compares the router IDs for identical eBGP paths. • **cost-community ignore**—Ignores the cost community for BGP best-path calculations. • **med confed**—Forces bestpath to do a MED comparison only between paths originated within a confederation. • **med missing-as-worst**—Treats a missing MED as the highest MED. • **med non-deterministic**—Does not always pick the best MED path from among the paths from the same autonomous system.
enforce-first-as **Example:** `switch(config-router)# enforce-first-as`	Enforces the neighbor autonomous system to be the first AS number listed in the AS_path attribute for eBGP.
log-neighbor-changes **Example**: `switch(config-router)# log-neighbor-changes`	Generates a system message when any neighbor changes state. **Note** To suppress neighbor status change messages for a specific neighbor, you can use the **log-neighbor-changes disable** command in router address-family configuration mode.
router-id *id* **Example**: `switch(config-router)# router-id 10.165.20.1`	Manually configures the router ID for this BGP speaker.

Command	Purpose
timers [**bestpath-delay** *delay* | **bgp** *keepalive holdtime* | **prefix-peer-timeout** *timeout*] **Example**: `switch(config-router)# timers bgp 90 270`	Sets the BGP timer values. The optional parameters are as follows: • *delay*—Initial best-path timeout value after a restart. The range is from 0 to 3600 seconds. The default value is 300. • *keepalive*—BGP session keepalive time. The range is from 0 to 3600 seconds. The default value is 60. • *holdtime*—BGP session hold time.The range is from 0 to 3600 seconds. The default value is 180. • *timeout*—Prefix peer timeout value. The range is from 0 to 1200 seconds. The default value is 30. You must manually reset the BGP sessions after configuring this command.

To tune BGP, use the following optional commands in router address-family configuration mode:

Command	Purpose
distance *ebgp-distance ibgp-distance local-distance* **Example**: `switch(config-router-af)# distance 20 100 200`	Sets the administrative distance for BGP. The range is from 1 to 255. The defaults are as follows: • *ebgp-distance*—20. • *ibgp-distance*—200. • *local-distance*—220. Local-distance is the administrative distance used for aggregate discard routes when they are installed in the RIB.
log-neighbor-changes [**disable**] **Example**: `switch(config-router-af)# log-neighbor-changes disable`	Generates a system message when this specific neighbor changes state. The **disable** option suppresses neighbor status change messages for this specific neighbor.

To tune BGP, use the following optional commands in neighbor configuration mode:

Command	Purpose
description *string* **Example**: `switch(config-router-neighbor)# description main site`	Sets a descriptive string for this BGP peer. The string can be up to 80 alphanumeric characters.
low-memory exempt **Example**: `switch(config-router-neighbor)# low-memory exempt`	Exempts this BGP neighbor from a possible shutdown due to a low memory condition.
transport connection-mode passive **Example:** `switch(config-router-neighbor)# transport connection-mode passive`	Allows a passive connection setup only. This BGP speaker does not initiate a TCP connection to a BGP peer. You must manually reset the BGP sessions after configuring this command.
[**no** | **default**] **remove-private-as** [**all** | **replace-as**] **Example**: `switch(config-router-neighbor)# remove-private-as`	Removes private AS numbers from outbound route updates to an eBGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions. The optional parameters are as follows: • **no**—Disables the command. • **default**—Moves the command to its default mode. • **all**—Removes all private-as numbers from the AS-path. • **replace-as**—Replaces all private AS numbers with the replace-as AS-path value. **Note** See the "Guidelines and Limitations for Advanced BGP" section for additional information on this command.
update-source *interface-type number* **Example**: `switch(config-router-neighbor)# update-source ethernet 2/1`	Configures the BGP speaker to use the source IP address of the configured interface for BGP sessions to the peer. This command triggers an automatic notification and session reset for the BGP neighbor sessions. Single-hop iBGP peers support fast external fallover when **update-source** is configured.

To tune BGP, use the following optional commands in neighbor address-family configuration mode:

Command	Purpose
allowas in **Example:** `switch(config-router-neighbor-af)# allowas in`	Allows routes that have their own AS in the AS path to be installed in the BRIB.
default-originate [**route-map** *map-name*] **Example:** `switch(config-router-neighbor-af)# default-originate`	Generates a default route to the BGP peer.
disable-peer-as-check **Example:** `switch(config-router-neighbor-af)# disable-peer-as-check`	Disables peer AS-number checking while the device advertises routes learned from one node to another node in the same AS path.
filter-list *list-name* {**in** | **out**} **Example:** `switch(config-router-neighbor-af)# filter-list BGPFilter in`	Applies an AS_path filter list to this BGP peer for inbound or outbound route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
prefix-list *list-name* {**in** | **out**} **Example:** `switch(config-router-neighbor-af)# prefix-list PrefixFilter in`	Applies a prefix list to this BGP peer for inbound or outbound route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
send-community **Example:** `switch(config-router-neighbor-af)# send-community`	Sends the community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
send-community extended **Example:** `switch(config-router-neighbor-af)# send-community extended`	Sends the extended community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
suppress-inactive **Example:** `switch(config-router-neighbor-af)# suppress-inactive`	Advertises the best (active) routes only to the BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

Configuring a Graceful Restart

You can configure a graceful restart and enable the graceful restart helper feature for BGP.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

Create the VRFs.

SUMMARY STEPS

1. **configure terminal**

2. **router bgp** *as-number*
3. **graceful-restart**
4. **graceful-restart** [**restart-time** *time* | **stalepath-time** *time*]
5. **graceful-restart-helper**
6. (Optional) **show running-config bgp**
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**router bgp** *as-number*
	
	Example:
	`switch(config)# router bgp 65535`
	`switch(config-router)#`
		Creates a new BGP process with the configured autonomous system number.

Step 3	**graceful-restart** **Example:** `switch(config-router)# graceful-restart`	Enables a graceful restart and the graceful restart helper functionality. This command is enabled by default. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 4	**graceful-restart** [**restart-time** *time* | **stalepath-time** *time*] **Example:** `switch(config-router)# graceful-restart restart-time 300`	Configures the graceful restart timers. The optional parameters are as follows: • **restart-time**—Maximum time for a restart sent to the BGP peer. The range is from 1 to 3600 seconds. The default is 120. • **stalepath-time**—Maximum time that BGP keeps the stale routes from the restarting BGP peer. The range is from 1 to 3600 seconds. The default is 300. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 5	**graceful-restart-helper** **Example:** `switch(config-router)# graceful-restart-helper`	Enables the graceful restart helper functionality. Use this command if you have disabled graceful restart but you still want to enable graceful restart helper functionality. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

	Command	Purpose
Step 6	**show running-config bgp** **Example:** switch(config-router)# show running-config bgp	(Optional) Displays the BGP configuration.
Step 7	**copy running-config startup-config** **Example:** switch(config-router)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to enable a graceful restart:

```
switch# configure terminal
switch(config)# router bgp 65535
switch(config-router)# graceful-restart
switch(config-router)# copy running-config startup-config
```

Configuring Virtualization

You can configure one BGP process, create multiple VRFs, and use the same BGP process in each VRF.

BEFORE YOU BEGIN

You must enable BGP (see the "Enabling BGP" section on page 9-11).

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **exit**
4. **router bgp** *as-number*
5. **vrf** *vrf-name*
6. **neighbor** *ip-address* **remote-as** *as-number*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**vrf context** *vrf-name* **Example:** switch(config)# vrf context RemoteOfficeVRF switch(config-vrf)#	Creates a new VRF and enters VRF configuration mode.
Step 3	**exit** **Example:** switch(config-vrf)# exit switch(config)#	Exits VRF configuration mode.
Step 4	**router bgp** *as-number* **Example:** switch(config)# router bgp 65535 switch(config-router)#	Creates a new BGP process with the configured autonomous system number.
Step 5	**vrf** *vrf-name* **Example:** switch(config-router)# vrf RemoteOfficeVRF switch(config-router-vrf)#	Enters the router VRF configuration mode and associates this BGP instance with a VRF.
Step 6	**neighbor** *ip-address* **remote-as** *as-number* **Example:** switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65535 switch(config-router--vrf-neighbor)#	Configures the IP address and AS number for a remote BGP peer.
Step 7	**copy running-config startup-config** **Example:** switch(config-router-vrf-neighbor)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to create a VRF and configure the router ID in the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# exit
switch(config)# router bgp 65535
switch(config-router)# vrf NewVRF
switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65535
switch(config-router-vrf-neighbor)# copy running-config startup-config
```

Verifying the Advanced BGP Configuration

To display the BGP configuration, perform one of the following tasks:

Command	Purpose
show bgp all [**summary**] [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp convergence [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **community** {**regexp** *expression* | [**community**] [**no-advertise**] [**no-export**] [**no-export-subconfed**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **community-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **extcommunity** {**regexp** *expression* | **generic** [**non-transitive** | **transitive**] *aa4:nn* [**exact-match**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **extcommunity-list** *list-name* [**exact-match**]} [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] {**dampening** **dampened-paths** [**regexp** *expression*]} [**vrf** *vrf-name*]	Displays the information for BGP route dampening. Use the **clear bgp dampening** command to clear the route flap dampening information.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **neighbors** [*ip-address* | *ipv6-prefix*] [**vrf** *vrf-name*]	Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] {**nexthop** | **nexthop-database**} [**vrf** *vrf-name*]	Displays the information for the BGP route next hop.
show bgp paths	Displays the BGP path information.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **policy** *name* [**vrf** *vrf-name*]	Displays the BGP policy information. Use the **clear bgp policy** command to clear the policy information.

Command	Purpose
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **route-map** *map-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the route map.
show bgp peer-policy *name* [**vrf** *vrf-name*]	Displays the information about BGP peer policies.
show bgp peer-session *name* [**vrf** *vrf-name*]	Displays the information about BGP peer sessions.
show bgp peer-template *name* [**vrf** *vrf-name*]	Displays the information about BGP peer templates. Use the **clear bgp peer-template** command to clear all neighbors in a peer template.
show bgp process	Displays the BGP process information.
show {**ipv4** | **ipv6**} **bgp** *options*	Displays the BGP status and configuration information.
show {**ipv4** | **ipv6**} **mbgp** *options*	Displays the BGP status and configuration information.
show running-configuration bgp	Displays the current running BGP configuration.

Monitoring BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
show bgp {**ipv4** | **ipv6**} {**unicast** | **multicast**} [*ip-address* | *ipv6-prefix*] **flap-statistics** [**vrf** *vrf-name*]	Displays the BGP route flap statistics.
show bgp {**ipv4** | **ipv6**} **unicast injected-routes**	Displays injected routes in the routing table.
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers.
show bgp statistics	Displays the BGP statistics.

Configuration Examples

This example shows how to configure MD5 authentication for prefix-based neighbors:

```
template peer BasePeer-V6
    description BasePeer-V6
    password 3 f4200cfc725bbd28
    transport connection-mode passive
    address-family ipv6 unicast
template peer BasePeer-V4
    bfd
    description BasePeer-V4
    password 3 f4200cfc725bbd28
    address-family ipv4 unicast
--
    neighbor fc00::10:3:11:0/127 remote-as 65006
      inherit peer BasePeer-V6
    neighbor 10.3.11.0/31 remote-as 65006
      inherit peer BasePeer-V4
```

This example shows how to enable neighbor status change messages globally and suppress them for a specific neighbor:

```
router bgp 65100
  log-neighbor-changes
    neighbor 209.165.201.1 remote-as 65535
      description test
      address-family ipv4 unicast
        soft-reconfiguration inbound
         disable log-neighbor-changes
```

Related Topics

The following topics can give more information on BGP:

- Chapter 9, “Configuring Basic BGP”
- Chapter 15, “Configuring Route Policy Manager”

Additional References

For additional information related to implementing BGP, see the following sections:

- MIBs, page 10-54

MIBs

MIBs	MIBs Link
MIBs related to BGP	To locate and download supported MIBs, go to the following URL: ftp://ftp.cisco.com/pub/mibs/supportlists/nexus9000/Nexus9000MIBSupportList.html