Configuring Advanced BGP

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

This chapter includes the following sections:

Information 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 routes and multiple Layer 3 protocol address families. BGP uses TCP as a reliable transport protocol to create TCP sessions with other BGP-enabled switches 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.

Peer Templates

BGP peer templates allow you to create blocks of common configurations 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 three mechanisms to reset BGP peering sessions:

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

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 for information on multihop, fast external failovers, and limiting the size of the AS-path attribute.

eBGP Next-Hop Unchanged

In an external BGP (eBGP) session, by default, the device changes the next-hop attribute of a BGP route (to its own address) when the device sends out a route. If the eBGP Next-Hop Unchanged feature is configured, BGP sends routes to an eBGP multihop peer without modifying the next-hop attribute. The next-hop attribute is unchanged. The BGP Next-hop Unchanged feature provides flexibility when designing and migrating networks. It can be used only between eBGP peers configured as multihop.

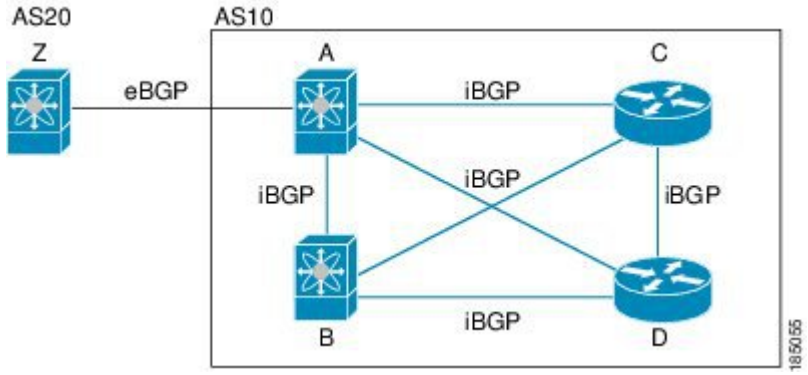
For example, consider a network with eBGP connection between Devices A, B, and C. Suppose Device A announces 100 prefixes to Device B. Device B is configured with an outbound route map to Device C and the match ip prefix list and set ip next-hop unchanged are configured on the route map. Device B propagates the unchanged next-hop address only for the routes that match the prefix list. For the other prefixes, it puts itself as the next-hop address.

iBGP

nternal 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).

The following figure shows an iBGP network within a larger BGP network.

Figure 1: iBGP Network



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

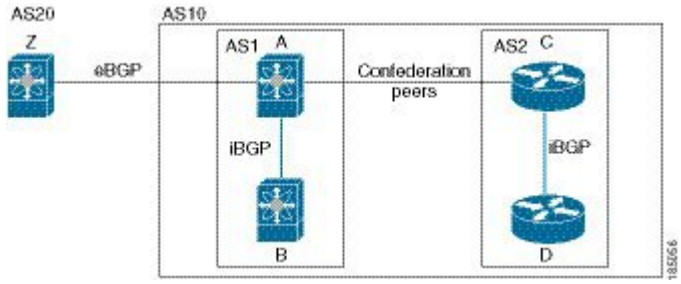
Note You should configure a separate interior gateway protocol in the iBGP network.

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.

The following figure shows the BGP network from Figure below, split into two subautonomous systems and one confederation.

Figure 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 **AS Confederation**.

Route Reflector

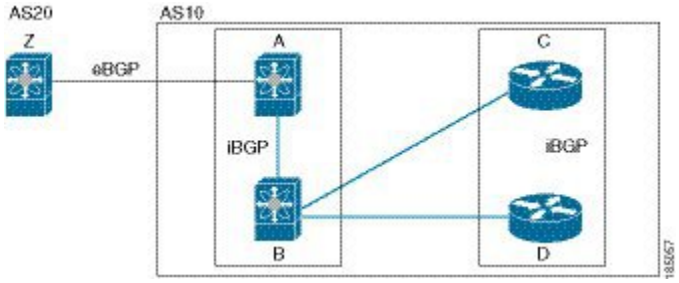
You can alternately reduce the iBGP mesh by using a route reflector configuration. Route reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

Figure **iBGP Network** shows a simple iBGP configuration with four meshed iBGP speakers (router 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 the following figure, 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 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 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 will attempt a new session to the peer without capabilities negotiation if you have configured the address family as IPv4.

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 selects only one of these multiple paths as the best path and advertises the path to the BGP peers.

Note Paths 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.

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 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 RIB that may affect BGP next-hop reachability.

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

- Next hop becomes unreachable.
- Next hop becomes reachable.
- Fully recursed IGP metric to the next hop changes.
- First hop IP address or first hop interface changes.
- Next hop becomes connected.
- Next hop becomes unconnected.
- Next hop becomes a local address.
- 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 the reachability (reachable and unreachable), connectivity (connected and unconnected), and locality (local and nonlocal) of the next hops. Notifications for these events are not delayed.
- Noncritical events include only the IGP metric changes.

See the Configuring BGP Next-Hop Address Tracking section for more information.

Site of Origin

The site of origin prevents routing loops when you have a multihomed VPN site. Routes learned from the same site are tagged with the same site-of-origin value that is configured at the PE on all the PE-CE links to the same site. Routes with a particular site-of-origin value are never readvertised back to a CE with the same site-of-origin value configured at the PE-CE link. This process prevents a CE router from relearning routes that originated from the same site. BGP and EIGRP use site of origin to prevent loops.

You can override the autonomous system number (ASN) of a site with the ASN of the provider. This feature is often used with the site of origin to identify the site where a route originated and prevent routing loops between routers within a VPN.

Route Redistribution

You can configure BGP to redistribute static routes or routes from other protocols. You configure a route policy with the redistribution to control which routes are passed into BGP. A route policy allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. See Configuring Route Policy Manager, for more information.

Labeled and Unlabeled Unicast Routes

In release 7.0(3)I7(6), SAFI-1 (unlabeled unicast) and SAFI-4 (labeled unicast routing) are now supported for IPv4 BGP on a single session. For more information, see the *Cisco Nexus 9000 Series NX-OS Label Switching Configuration Guide, Release 7.x*.

BFD

This feature supports bidirectional forwarding detection (BFD). 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 single-hop peers and iBGP single-hop peers. For iBGP single-hop peers using BFD, you must configure the update-source option in neighbor configuration mode. BFD is not supported on other iBGP peers or for multihop eBGP peers.

Beginning with Cisco NX-OS Release 9.3(3), BFD for BGP is also supported for BGP IPv4 and IPv6 prefix peers. This support enables BGP to use multihop BFD, which improves BGP convergence times. Both single-hop and multihop BGP are supported for prefix peers.

BFD is supported for the following types of interfaces:

- Layer 3 physical and subinterface

- Layer 3 port channel and subinterface
- Switch virtual interface (SVI)

BFD for BGP does not support authentication or per-link BFD sessions on a port channel.

See Configuring Bidirectional Forwarding Detection for more information.

Tuning BGP

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

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 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 and one set of routes for IPv4 multicast routing. You can use MP-BGP for reverse-path forwarding (RPF) checks in IP multicast networks.

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.

BGP Monitoring Protocol

The BGP Monitoring Protocol (BMP) monitors BGP updates and peer statistics. BMP is supported for all the Cisco Nexus 3000 Series switches.

Using this protocol, the BGP speaker connects to external BMP servers and sends them information regarding BGP events. A maximum of two BMP servers can be configured in a BGP speaker, and each BGP peer can

be configured for monitoring by all or a subset of the BMP servers. The BGP speaker does not accept any information from the BMP server.

RFC 5549

Beginning with Cisco NX-OS Release 6.0(2)U4(1), BGP supports RFC 5549, which allows an IPv4 prefix to be carried over an IPv6 next hop. Because BGP is running on every hop, and all routers are capable of forwarding IPv4 and IPv6 traffic, there is no need to support IPv6 tunnels between any routers. BGP installs IPv4 over an IPv6 route to the Unicast Route Information Base (URIB)

Prerequisites for Advanced BGP

BGP has the following prerequisites:

- You must enable the BGP feature (see the Enabling the BGP Feature section).
- 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

BGP has the following configuration guidelines and limitations:

- Prefix peering operates only in passive TCP mode. It accepts incoming connections from remote peers if the peer address falls within the prefix.
- 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 redistribution.
- Configure the BGP router ID within a VRF.
- If you decrease the keepalive and hold timer values, the network might experience session flaps.

- The following guidelines and limitations apply to the **remove-private-as** command:
  - If the local AS number of the device is a private AS number, you cannot use the remove-private-as configuration command for any other neighbor on the same device. As a workaround, you can use the local-as command on each neighbor with a public local AS number.
  - If the real AS number of the device is a private AS number and the remove-private-as all command is configured for a neighbor with a public local-as number, use local-as number [no-prepend [replace-as]] command to ensure that the real private AS number is not appended to the AS path.
  - If the real AS number of the device is a public AS number and the remove-private-as all command is configured for a neighbor, you cannot configure a private local-as number for the same neighbor. As a workaround, you must remove the existing configuration to proceed further.
  - The remove-private-as all command removes private AS numbers from the AS path even if the path contains both public and private AS numbers.
  - The remove-private-as command removes private AS numbers even if the AS path contains only private AS numbers. There is no likelihood of a 0-length AS path because this command can be applied to eBGP peers only, in which case the AS number of the local device is appended to the AS path.
  - The remove-private-as command removes private AS numbers even if the private AS numbers appear before the confederation segments in the AS path.
  - When you remove private AS numbers from the AS path, the path length of the prefixes that are sent out will decrease. Because the AS path length is a key element of BGP best-path selection, it might be necessary to retain the path length. The replace-as keyword ensures that the path length is retained by replacing all removed AS numbers with the local router's AS number.
  - On Cisco Nexus 3000 Series switches running Cisco NX-OS releases 7.0(3)I7(3) and 7.0(3)I7(4), a fragmented BGP Update packet that has MD5 authentication may not be properly reassembled upon reception. This situation can prevent the BGP session from coming up in a stable state. When this condition occurs, an error message similar to the following is displayed:

    ```
    %KERN-3-SYSTEM_MSG: [322896.818155] TCP: MD5 Hash failed for (10.1.1.1,
    179)->(10.2.2.2, 29862) - kernel
    %KERN-3-SYSTEM_MSG: [322897.020543] TCP: MD5 Hash failed for (10.1.1.1,
    179)->(10.2.2.2, 29862) - kernel
    ```

    To prevent this situation, do either of the following:
    - Disable BGP MD5 authentication.
    - Avoid fragmentation of the BGP packets by setting the MTU value on the outbound interface to match the MTU of the path.

- Beginning with Cisco NX-OS Release 9.3(3), BFD for BGP is supported for BGP IPv4 and IPv6 prefix peers.
- Beginning with Cisco NX-OS Release 9.3(3), BGP prefix peers support graceful restarts. You can use the **timers prefix-peer-timeout** command in router configuration mode to configure the timeout value (in seconds) for BGP prefix peers. The default value is 90 seconds.
- The following guidelines and limitations apply to BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:
  - This feature does not support having the same link-local address configured across multiple interfaces.

- This feature is not supported on logical interfaces (loopback). Only Ethernet interfaces, port-channel interfaces, subinterfaces, and breakout interfaces are supported.
- Beginning with Cisco NX-OS Release 9.3(6), VLAN interfaces are supported.
- This feature is supported only for IPv6-enabled interfaces with link-local addresses.
- This feature is not supported when the configured prefix peer and interface have the same remote peer.
- The following commands are not supported in neighbor interface configuration mode:
  - **disable-connected-check**
  - **maximum-peers**
  - **update-source**
  - **ebgp-multihop**
- BFD multihop and the following commands are not supported for BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:
  - **bfd-multihop**
  - **bfd multihop interval**
  - **bfd multihop authentication**
- BGP requires faster convergence time for route advertisements. To speed up detection of the Route Advertisement (RA) link-level protocol, enter the following commands on each IPv6-enabled interface that is using BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:

```
interface Ethernet port/slot
ipv6 nd ra-interval 4 min 3
ipv6 nd ra-lifetime 10
```

Default Settings for BGP

Table below lists the default settings for BGP parameters.

Table 1: Default BGP Parameters

Parameters	Default
BGP feature	disabled
keep alive interval	60 seconds
hold timer	180 seconds

Configuring Advanced BGP

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.

Enabling IP Forward on an Interface

To use RFC 5549, you must configure at least one IPv4 address. If you do not want to configure an IPv4 address, you must enable the ip forward feature to use RFC 5549.

Procedure

	Command or Action	Purpose

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

Step 2	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**ip forward**
	Example:
	`switch(config-if)# ip forward`
	`switch(config-if)#`
		Allows IPv4 traffic on the interface even when there is no IP address configuration on that interface.

Example

This example shows how to enable the ip forward feature on an interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip forward
```

Configuring BGP Session Templates

You can use BGP session templates to simplify 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

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

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.

Procedure

	Command or Action	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 65536`
	`switch(config-router)#`
		Enables BGP and assigns the autonomous system number to the local BGP speaker.

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	(Optional) **password** *number password* **Example:** `switch(config-router-stmp)# password 0 test`	Adds the cleartext password *test* to the neighbor. The password is stored and displayed in type 3 encrypted form (3DES).
Step 5	(Optional) **timers** *keepalive hold* **Example:** `switch(config-router-stmp)# timers 30 90`	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.

	Command or Action	Purpose

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65536`
	`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	(Optional) **description** *text*
	Example:
	`switch(config-router-neighbor)# description Peer Router A`
	`switch(config-router-neighbor)`
		Adds a description for the neighbor.

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

Example

Use the **show bgp neighbor** command to see the template applied. See the Cisco Nexus 3000 Series Command Reference for details on all commands available in the template.

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 65536
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 65536
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

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

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.

Procedure

	Command or Action	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 65536`
	`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	(Optional) **advertise-active-only** **Example:** `switch(config-router-ptmp)# advertise-active-only`	Advertises only active routes to the peer.
Step 5	(Optional) **maximum-prefix** *number* **Example:**	Sets the maximum number of prefixes allowed from this peer.

	Command or Action	Purpose
	`switch(config-router-ptmp)# maximum-prefix 20`	

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 65536`
	`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.
Step 10	(Optional) **show bgp peer-policy** *template-name* **Example:** `switch(config-router-neighbor-af)# show bgp peer-policy BasePolicy`	Displays the peer-policy template.
Step 11	(Optional) copy running-config startup-config **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **show bgp neighbor** command to see the template applied. See the Cisco Nexus 3000 Series Command Reference for details on all commands available in the template.

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 65536
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 65536
```

```
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

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

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.

Procedure

	Command or Action	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 65536`	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	(Optional) **inherit peer-session** *template-name* **Example:** `switch(config-router-neighbor)# inherit peer-session BaseSession`	Inherits a peer-session template in the peer template.

	Command or Action	Purpose

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

Step 6	(Optional) **inherit peer** *template-name* **Example:** `switch(config-router-neighbor-af)# inherit peer BasePolicy`	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.

Step 8	(Optional) **timers** *keepalive hold* **Example:** `switch(config-router-neighbor)# timers 45 100`	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 65536`
	`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	(Optional) **timers** *keepalive hold* **Example:** `switch(config-router-neighbor)# timers 60 120`	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	(Optional) **show bgp peer-template** *template-name* **Example:** `switch(config-router-neighbor-af)# show bgp peer-template BasePeer`	Displays the peer template.

	Command or Action	Purpose
Step 14	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

Use the **show bgp neighbor** command to see the template applied. See the Cisco Nexus 3000 Series Command Reference for details on all commands available in the template.

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

```
switch# configure terminal
switch(config)# router bgp 65536
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 65536
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 router configuration mode:

Command	Purpose
timers prefix-peer-timeout *value* **Example** : `switch(config-router)# 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. **Note** For prefix peers, set the prefix peer timeout to be greater than the configured graceful restart timer. If the prefix peer timeout is greater than the graceful restart timer, a peer's route is retained during its restart. If the prefix peer timeout is less than the graceful restart timer, the peer's route is purged by the prefix peer timeout, which may occur before the restart is complete.

Command	Purpose
timers prefix-peer-wait interval **Example** : `switch(config-router)# timers prefix-peer-wait 50`	Configures the BGP prefix peering wait timer on a per-VRF basis or on the default VRF. You can use the timers prefix-peer-wait command to disable the peer prefix wait time so that there is no delay before BGP prefixes are inserted into the routing information base (RIB). The range of the interval is from 0 to 1200 seconds. The default value is 90 seconds. **Note** The timer is only applicable for BGP dynamic neighbors. It is only set when BGP is restarted or is coming up for the first time for dynamic BGP neighbors.

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(config)# router bgp 65536
switch(config-router)# timers prefix-peer-timeout 120
switch(config-router)# neighbor 10.100.200.0/24 remote-as 65536
switch(config-router-neighbor)# maximum-peers 10
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

This example shows how to disable the peer prefix wait time:

```
switch(config)# router bgp 100
switch(config-router)# timers prefix-peer-wait 50
switch(config-router)#
```

Use the **show ip bgp 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 Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families

You can configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families for automatic BGP neighbor discovery using unnumbered interfaces. Doing so allows you to set up BGP sessions using an interface name as a BGP peer (rather than interface-scoped addresses). This feature relies on ICMPv6 neighbor discovery (ND) route advertisement (RA) for automatic neighbor discovery and on RFC 5549 for sending IPv4 routes with IPv6 next hop.

Before you begin

You must enable BGP.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters 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. 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.

Step 3	**neighbor** *interface-name* **remote-as** {*as-number* | **route-map** *map-name*}
	Example:
	`switch(config-router)# neighbor Ethernet1/1 remote-as route-map Testmap`
	`switch(config-router-neighbor)#`
		Places the router in the neighbor configuration mode for BGP routing and configures the interface for BGP peering.
		Note You can specify only Ethernet interfaces, port-channel interfaces, subinterfaces, and breakout interfaces.
		Beginning with Cisco NX-OS Release 9.3(6), you can specify a route map, which can contain AS lists and ranges. See Dynamic AS Numbers for Prefix Peers for more information about using dynamic AS numbers.

Step 4	**inherit peer** *template-name* **Example:** `switch(config-router-neighbor)# inherit peer PEER`	Inherits the peer template.

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

Step 6	(Optional) **show bgp** {**ipv4** | **ipv6**} **unicast neighbors** *interface* **Example:** `switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors e1/25`	Displays information about BGP peers.

	Command or Action	Purpose
	Example: `switch(config-router-neighbor-af)# show bgp ipv6 unicast neighbors 3FFE:700:20:1::11`	
Step 7	(Optional) **show ip bgp neighbors** *interface-name* **Example:** `switch(config-router-neighbor-af)# show ip bgp neighbors Ethernet1/1`	Displays the interface used as a BGP peer.
Step 8	(Optional) **show ipv6 routers** [**interface** *interface*] **Example:** `switch(config-router-neighbor-af)# show ipv6 routers interface Ethernet1/1`	Displays the link-local address of remote IPv6 routers, which is learned through IPv6 ICMP router advertisement.
Step 9	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families using a route map:

iBGP Interface Peering Configuration for Leaf 1:

```
switch# configure terminal
switch(config)# route-map Testmap permit 10
switch(config-route-map)# match as-number 100-200, 300, 400
switch(config-route-map)# exit
switch(config)# router bgp 65000
switch(config-router)# neighbor Ethernet1/1 remote-as route-map Testmap
switch(config-router-neighbor)# inherit peer PEER
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor)# address-family ipv6 unicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

This example shows sample output for BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families:

```
switch(config-router-neighbor)# show bgp ipv4 unicast neighbors e1/15.1
BGP neighbor is fe80::2, remote AS 100, ibgp link, Peer index 4
Peer is an instance of interface peering Ethernet1/15.1
BGP version 4, remote router ID 5.5.5.5
Neighbor previous state = OpenConfirm
BGP state = Established, up for 2d16h
Neighbor vrf: default
Peer is directly attached, interface Ethernet1/15.1
Last read 00:00:54, hold time = 180, keepalive interval is 60 seconds
```

```
Last written 00:00:08, keepalive timer expiry due 00:00:51
Received 3869 messages, 0 notifications, 0 bytes in queue
Sent 3871 messages, 0 notifications, 0(0) bytes in queue
Enhanced error processing: On
0 discarded attributes
Connections established 2, dropped 1
Last reset by peer 2d16h, due to session closed
Last error length received: 0
Reset error value received 0
Reset error received major: 104 minor: 0
Notification data received:
Last reset by us never, due to No error
Last error length sent: 0
Reset error value sent: 0
Reset error sent major: 0 minor: 0
--More--
```

Interface Configuration:

IPv6 needs to be enabled on the corresponding interface using one of the following commands:

- **ipv6 address** *ipv6-address*
- **ipv6 address use-link-local-only**
- **ipv6 link-local** *link-local-address*

```
switch# configure terminal
switch(config)# interface Ethernet1/1
switch(config-if)# ipv6 address use-link-local-only
```

Note If an IPv4 address is not configured on the interface, the **ip forward** command must be configured on the interface to enable IPv4 forwarding.

Note IPv6 ND timers can be tuned to speed up neighbor discovery and for BGP faster route convergence.

```
switch(config-if)# ipv6 nd ra-interval 4 min 3
switch(config-if)# ipv6 nd ra-lifetime 10
```

Note Beginning with Cisco NX-OS Release 9.3(6), for customer deployments with parallel links, the following command must be added in interface mode:

```
switch(config-if)# ipv6 link-local use-bia
```

The command makes IPv6 LLA unique across different interfaces.

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 always **Example** : `switch(config-router-neighbor-af)# soft-reconfiguration inbound always`	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 ip { **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 the 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 parameters in commands 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.

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 RIB that may affect BGP next-hop reachability.

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

- Next hop becomes unreachable.
- Next hop becomes reachable.
- Fully recursed IGP metric to the next hop changes.
- First hop IP address or first hop interface changes.
- Next hop becomes connected.
- Next hop becomes unconnected.
- Next hop becomes a local address.
- 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 the reachability (reachable and unreachable), connectivity (connected and unconnected), and locality (local and nonlocal) of the next hops. Notifications for these events are not delayed.
- Noncritical events include only the IGP metric changes.

See the Configuring BGP Next-Hop Address Tracking section for more information.

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.

Controlling Reflected Routes Through Next-Hop-Self

NX-OS enables controlling the iBGP routes being sent to a specific peer through the **next-hop-self** [all] arguments. By using these arguments, you can selectively change the next-hop of routes even if the route is reflected.

Command	Purpose
next-hop-self [all] **Example**: `switch(config-router-af)# next-hop-self all`	Uses the local BGP speaker address as the next-hop address in route updates. The all keyword is optional. If you specify all, all routes are sent to the peer with next-hop-self. If you do not specify all, the next hops of reflected routes are not changed.

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
- BFD failure detections for BGP neighbors
- Administrative shutdown of BGP neighbors (using the shutdown command)

The accelerated handling of Layer 3 link 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 the 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 and 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.

Configuring eBGP

This section includes the following topics:

Configuring eBGP Next-Hop Unchanged

You can configure eBGP to send routes to an eBGP multihop peer without changing the next-hop address. By default, the device changes the next-hop address of a BGP route to its own address when the device sends out a route.

	Command	Purpose
Step 1	**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.
Step 2	**configure terminal** Example: `switch# configure terminal`	Enters global configuration mode.

	Command	Purpose
Step 3	**route-map** *route-map name* Example: `switch(config)# route-map route`	Enters route map configuration mode.
Step 4	**set ip next-hop unchanged** Example: `switch(config-route-map)# set ip next-hop unchanged`	Configures the device to send BGP updates to the specified eBGP peer without modifying the next-hop address.
Step 5	**exit** Example: `switch(config-route-map)# exit`	Exits route map configuration mode.

This example shows how to set eBGP next-hop unchanged to send routes without changing the next-hop address:

```
switch# configure terminal
switch(config)# route-map route
switch(config-route-map)# set ip next-hop unchanged
switch(config-route-map)# exit
switch(config)#
```

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.

Configuring eBGP Routes in the Same Autonomous System

You can configure eBGP learned routes from a remote autonomous system (AS) to advertise to another eBGP peer in the same AS.

Note When route updates are sent between peers within the same AS number, they are dropped unless you enter the **allowas-in** command.

To disable AS peer checking, use the following command in neighbor configuration mode:

	Command	Purpose
Step 1	**router bgp** *autonomous-system-number* **Example** : `switch(config)# router bgp 64496`	Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.
Step 2	**neighbor ipv4 remote-as** *as-number* **Example** : `switch (config-router)# neighbor 209.165.201.1 remote-as 64497`	Configures the specified address type and AS number for a remote BGP peer. The ip-address format is x.x.x.x. The IPv6 address-format is A:B::C:D.
Step 3	**address family ipv4 unicast** **Example** : `switch(config-router)# address family ip4 unicast`	Enters neighbor address family configuration mode for the unicast specified address family.
Step 4	**disable-peer-as-check** **Example** : `switch(config-router-neighbor-af)# disable-peer-as-check`	Disables AS checking so that routes are updated between peers in the same AS.
Step 5	**show bgp neighbor** **Example** : `switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors`	Displays information about BGP peers.

This example shows how to display BGP peer information:

```
switch(config)# show bgp neighbor 1.222.222.2
bgp neighbor is 1.222.222.2, remote as 2222, ebgp link, peer index 1
bgp version 4, remote router id1.100.1.2 ####output trunicated####
for address family:ipv4 unicast
bgp table version 54, neighbor version 54
3 accepted paths consume 108 bytes of memory
10 sent paths
peer asn check is disabled

#####output omitted#####
```

Disabling a Fast External Failover

Typically, when a BGP router loses connectivity to a directly connected eBGP peer, BGP triggers a fast external failover by resetting the eBGP session to the peer. You can disable this fast external failover to limit the instability caused by link flaps.

To disable a fast external failover, 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 failover for eBGP peers. This command is enabled by default.

Limiting the AS-path Attribute

You can configure eBGP to discard routes that have a high number of AS numbers in the AS-path attribute.

To discard routes that have a high number of 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.

Local AS 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 **local-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. The **no-prepend** keyword ensures that the **local-as** *number* is not prepended to any downstream BGP neighbors except for the partner who is peering with the **local-as** *number*. The **replace-as** keyword ensures that only the **local-as** *number* of the peering session is prepended to the AS_PATH attribute. The autonomous-system number from the local BGP routing process is not prepended. The **dual-as** keyword configures the eBGP neighbor to establish a peering session using the real autonomous-system number (from the local BGP routing process) or by using the autonomous-system number configured as the Local AS).

Configuring AS Confederations

To configure an AS confederation, you must specify a confederation identifier. The group of autonomous systems within the AS confederation looks 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 64512`	Configures a confederation identifier for an AS confederation. Each confederation has a different sub-AS number, usually a private one (from 64512 to 65534). 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 an Autonomous System Path Containing Your Own Autonomous System

Enable the feature for BGP to accept the autonomous system (AS) path that contains your own AS.

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *as-number*	Enters BGP mode and assigns the autonomous system number to the local BGP speaker. The *as-number* value range is from 1 to 65535.
Step 3	**neighbor** *ip-address* **remote-as** *as-number*	Enters neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 4	**address-family** { **ipv4** | **ipv6**} { **multicast** | **unicast** }	Enters router address family configuration for the specified address family.
Step 5	[**no** | **default**] **allowas-in** [*allowas-in-cnt*]	Enables the allowas-in feature for BGP and configures the number of occurrences of the AS number. For **allowas-in-cnt**, enter an integer between 1 and 10. By default, the number of occurrences of the AS number is set to 3.
Step 6	**end**	Exits router address family configuration mode.
Step 7	(Optional) **show running-config bgp**	Displays the BGP configuration.
Step 8	**copy running-config startup-config**	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Example

This example shows how to configure the allowas-in feature for BGP and configure it for a unicast IPv4 address family:

```
switch# configure terminal
switch(config)# router bgp 77
switch(config-router)# neighbor 6.20.1.1 remote-as 66
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# allowas-in 5
switch(config-router-neighbor-af)# end
```

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

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	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 65536`
	`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	(Optional) **client-to-client reflection** **Example:** `switch(config-router-af)# client-to-client reflection`	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.

	Command or Action	Purpose

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.0.2.10 remote-as 65536`
	`switch(config-router-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

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 switch 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	(Optional) **show bgp ip** { **unicast** | **multicast** } **neighbors** **Example:** `switch(config-router-neighbor-af)# show bgp ip unicast neighbors`	Displays the BGP peers.
Step 11	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

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

```
switch(config)# router bgp 65536
switch(config-router)# neighbor 192.0.2.10 remote-as 65536
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 Route Dampening

You can configure route dampening to minimize eBGP 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 Route Deletion Delay

When routes are deleted from the hardware faster than the BGP neighbor can remove them from the next-hop list, the traffic that is received for those routes is lost. You can delay route deletion from the hardware by configuring a route deletion interval.

This configuration is not applied when a subset of next hops is being removed from the route or when a backup path exists for the route. It applies only to a route that is completely getting deleted.

Command	Purpose

route delete dampen interval [*seconds*]
Example :
`switch(config)# route delete dampen interval 20`
`switch(config)#`
	Delays route deletion from the hardware.
	The interval can range from 1 second to 30 seconds.
	The default value is 0 seconds.

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 12`	Configures the maximum number of equal-cost paths for load sharing. The range is from 1 to 16. 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* | **warming-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. This command is enabled by default.

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 filters 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. BGP route suppression is not enabled by default.

Note When you enable fib-suppression on the switch for routes that are not programmed locally in the hardware because of hardware table exhaustion, BGP advertises these failed routes even though they are not programmed locally in the hardware.

To suppress BGP routes, use the following command in the 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

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	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 65536`
	`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 65537`
	`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 will advertise

	Command or Action	Purpose
		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 will advertise the route. The nonexist-rmap is a case-sensitive, alphanumeric string up to 63 characters.
Step 6	(Optional) **show ip bgp neighbor** **Example:** `switch(config-router-neighbor-af)# show ip bgp neighbor`	Displays information about BGP and the configured conditional advertisement route maps.
Step 7	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to configure BGP conditional advertisement:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# neighbor 192.0.2.2 remote-as 65537
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 209.165.201.0/27
```

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

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	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 65536`
	`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** | **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 for more information about route maps.
Step 5	(Optional) **default-metric** *value* **Example:** `switch(config-router-af)# default-metric 33`	Generates a default route into BGP.
Step 6	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-af)# copy running-config startup-config`	Saves this configuration change.

Example

This example shows how to redistribute EIGRP into BGP:

```
switch# configure terminal
switch(config)# router bgp 65536
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
```

Disabling BGP Dampening with Redistribution

When an IGP metric of routes redistributed into BGP changes, BGP has internal dampening that prevents an immediate route update to the BGP peers. It affects how BGP handles IGP metric changes reported for redistributed routes. BGP dampens these changes through a batch process with a 10-minute delay. This command enables you to adjust that delay or remove it altogether for a quicker response to these changes.

Procedure

	Command or Action	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 65536`
	`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	**dampen-igp-metric** *seconds* **Example:** `switch(config-router-af)# dampen-igp-metric 100`	Configures dampening of IGP metric-related changes for redistributed routes.

Example

This example shows how to configure BGP dampening for redistributed routes:

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

Configuring Multiprotocol BGP

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

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	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 65536`
	`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 65537`
	`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	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	Saves this configuration change.

Example

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

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ipv6 address 2001:0DB8::1
switch(config-if)# router bgp 65536
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
```

Configuring BGP Extended Community Site of Origin

To configure BGP extended community site of origin, use the following commands

Command	Purpose

router bgp *as-number*
Example :
`switch(config)# router bgp 1`
`switch(config-router)#`
	Configures a BGP routing process and enters router configuration mode.

vrf *vrf-name*
Example:
`switch(config-router)# vrf 450`
`switch(config-router-vrf)#`
	Enters the router VRF configuration mode and associates this BGP instance with a VRF.

neighbor ip-address remote-as *as-number*
Example:
`switch(config-router-vrf)# neighbor 1::1 remote-as 2`
`switch(config-router-vrf-neighbor)#`
	Configures the IP address and AS number for a remote BGP peer.

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

soo value Example: `switch(config-router-vrf-neighbor-af)# soo 22:14`	Configures the site of origin BGP extended community value. The value is in one of the following formats: • asn:number • IP address:number The number range is from 0 to 65535 for a 2-byte ASN or from 0 to 4294967295 for a 4-byte ASN.

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** | **compare-routerid** | **igp-metric ignore** | **med** { **missing-as-worst** | **non-deterministic** }| **as-path multipath-relax**]
Example :
`switch(config-router)# bestpath always-compare-med`
`switch(config-router)# bestpath as-path multipath-relax`
	Modifies the best-path algorithm. The optional parameters are as follows:
	• always-compare-med —Compares MED on paths from different autonomous systems.
	• compare-routerid—Compares the router IDs for identical eBGP paths.
	• igp-metric ignore—Ignores the Interior Gateway Protocol (IGP) metric for next hop during best-path selection. This option is supported beginning with Cisco NX-OS Release 9.2(2).
	• med missing-as-worst—Handles a missing MED as the highest MED.
	• med non-deterministic —Does not always select the best MED path from among the paths from the same autonomous system.
	• as-path multipath-relax —Allows the switch to handle the paths received from different AS numbers for multipath, if their AS-path lengths are the same and other multipath conditions are met.

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 a neighbor changes state. **Note** To suppress neighbor status change messages for a specific neighbor, you can use the log-neighbor-changes disable command in neighbor configuration mode.
router-id *id* **Example** : `switch(config-router)# router-id 209.165.20.1`	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.
bestpath all-paths-ecmp **Example** : `switch(config-router-af)# bestpath all-paths-ecmp`	Treats all paths as ECMP during best path calculation

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.

Command	Purpose
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.
remove-private-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. **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 1/2`	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.
log-neighbor-changes [**disable**] **Example** : `switch(config-router-neighbor)# 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 address-family configuration mode:

Command	Purpose
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.
default-originate [**route-map** *map-name*] **Example** : `switch(config-router-neighbor-af)# default-originate`	Generates a default route to the BGP peer.
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.

Command	Purpose
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-extcommunity **Example** : `switch(config-router-neighbor-af)# send-extcommunity`	Sends the extended community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
[no | default] as-override **Example:** `switch(config-router-neighbor-af)# as-override`	**no** - (Optional) Disables the command. **default** - (Optional) Moves the command to its default mode. **as-override** - While sending updates to eBGP peer, replaces in the *path* attribute all occurrences of the peer's AS number with the local AS number.

Configuring Virtualization

Before you begin

Ensure that you have enabled the BGP feature (see the Enabling the BGP Feature section).

Procedure

	Command or Action	Purpose

Step 1	**configure terminal**
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters 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:**	Creates a new BGP process with the configured autonomous system number.

	Command or Action	Purpose
	`switch(config)# router bgp 65536` `switch(config-router)#`	

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 65536`
	`switch(config-router--vrf-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

Step 7	(Optional) **bestpath as-path multipath-relax** **Example:** `switch(config-router-vrf)# bestpath as-path multipath-relax`	Allows the switch to treat paths received from different autonomous systems for multipath, if their autonomous path lengths are the same and other multipath conditions are met.
Step 8	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	Saves this configuration change.

Example

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 65536
switch(config-router)# vrf NewVRF
switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65536
switch(config-router-vrf-neighbor)# copy running-config startup-config
```

Configuring BMP

You can configure BMP on the device.

Before you begin

You must enable BGP.

Procedure

	Command or Action	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	**bmp server server-number**
	Example:
	`switch(config-router)# bmp server 1`
	`switch(config-router)#`
		Configures the BMP server to which BGP should send information. The server number is used as a key.
		Note You can configure up to two BMP servers

Step 4	**address** *ip-address* **port-number** *port-number* **Example:** `switch(config-router)# address 10.1.1.1 port-number 2000`	Configures the IPv4 or IPv6 address of the host and the port number on which the BMP speaker connects to the BMP server.
Step 5	**description** *string* **Example:** `switch(config-router)# description BMPserver1`	Configures the BMP server description. You can enter up to 256 alphanumeric characters.
Step 6	**initial-refresh** { **skip** | **delay** *time* } **Example:** `switch(config-router)# initial-refresh delay 100`	Configures the option to send a route refresh when BGP is converged and the BMP server connection is established later. The skip option specifies to not send a route refresh if the BMP server connection comes up later. The delay option specifies the time in seconds after which the route refresh should be sent. The range is from 30 to 720 seconds, and the default value is 30 seconds.
Step 7	**initial-delay** *time* **Example:** `switch(config-router)# initial-delay 120`	Configures the delay after which a connection is attempted to the BMP server. The range is from 30 to 720 seconds, and the default value is 45 seconds.
Step 8	**stats-reporting-period** *time* **Example:**	Configures the time interval in which the BMP server receives the statistics report from BGP

	Command or Action	Purpose
	`switch(config-router)# stats-reporting-period 50`	neighbors. The range is from 30 to 720 seconds, and the default is disabled.
Step 9	**shutdown** **Example:** `switch(config-router)# shutdown`	Disables the connection to the BMP server.
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)#`	Enters neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 11	**bmp-activate-server** *server-number* **Example:** `switch(config-router-neighbor)# bmp-activate-server 1`	Configures the BMP server to which a neighbor's information should be sent.
Step 12	(Optional) **show bgp bmp server** [*server-number*][*detail*] **Example:** `switch(config-router-neighbor)# show bgp bmp server`	Displays BMP server information.
Step 13	(Optional) **copy running-config startup-config** **Example:** `switch(config-router-neighbor)# copy running-config startup-config`	Saves this configuration change.

Configuring BGP Attribute Filtering and Error Handling

Beginning with Cisco NX-OS Release 9.3(3), you can configure BGP attribute filtering and error handling to provide an increased level of security. The following features are available and implemented in the following order:

- **Path attribute treat-as-withdraw:** Allows you to treat-as-withdraw a BGP update from a specific neighbor if the update contains a specified attribute type. The prefixes contained in the update are removed from the routing table.
- **Path attribute discard:** Allows you to remove specific path attributes in a BGP update from a specific neighbor.
- **Enhanced attribute error handling:** Prevents peer sessions from flapping due to a malformed update.

Attribute types 1, 2, 3, 4, 5, 8, 14, 15, and 16 cannot be configured for path attribute treat-as-withdraw and path attribute discard. Attribute type 9 (Originator) and type 10 (Cluster-id) can be configured for eBGP neighbors only.

Treating as Withdraw Path Attributes from a BGP Update Message

To "treat-as-withdraw" BGP updates that contain specific path attributes, use the following command in router neighbor configuration mode:

Procedure

	Command or Action	Purpose

Step 1	[**no**] **path-attribute treat-as-withdraw** [*value* | **range** *start end*] **in**
	Example:
	`switch#(config-router)# neighbor 10.20.30.40`
	`switch(config-router-neighbor)# path-attribute treat-as-withdraw 100 in`
	Example:
	`switch#(config-router)# neighbor 10.20.30.40`
	`switch(config-router-neighbor)# path-attribute treat-as-withdraw range 21 255 in`
		Treats as withdraw any incoming BGP update messages that contain the specified path attribute or range of path attributes and triggers an inbound route refresh to ensure that the routing table is up to date. Any prefixes in a BGP update that are treat-as-withdraw are removed from the BGP routing table.
		This command is also supported for BGP template peers and BGP template peer sessions.

Discarding Path Attributes from a BGP Update Message

To discard BGP updates that contain specific path attributes, use the following command in router neighbor configuration mode:

Procedure

	Command or Action	Purpose

Step 1	[**no**] **path-attribute discard** [*value* | **range** *start end*] **in**
	Example:
	`switch#(config-router)# neighbor 10.20.30.40`
	`switch(config-router-neighbor)# path-attribute discard 100 in`
	Example:
	`switch#(config-router)# neighbor 10.20.30.40`
	`switch(config-router-neighbor)# path-attribute discard range 100 255 in`
		Drops specified path attributes in BGP update messages for the specified neighbor and triggers an inbound route refresh to ensure that the routing table is up to date. You can configure a specific attribute or an entire range of unwanted attributes.
		This command is also supported for BGP template peers and BGP template peer sessions.
		Note When the same path attribute is configured for both discard and treat-as-withdaw, treat-as-withdraw has a higher priority.

Enabling or Disabling Enhanced Attribute Error Handling

BGP enhanced attribute error handling is enabled by default but can be disabled. This feature, which complies with RFC 7606, prevents peer sessions from flapping due to a malformed update. The default behavior applies to both eBGP and iBGP peers.

To disable or reenable enhanced error handling, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose

Step 1	[**no**] **enhanced-error**
	Example:
	`switch(config)# router bgp 1000`
	`switch(config-router)# enhanced-error`
		Enables or disables BGP enhanced attribute error handling.

Displaying Discarded or Unknown Path Attributes

To display information about discarded or unknown path attributes, perform one of the following tasks:

Command	Purpose
show bgp {**ipv4** | **ipv6**} **unicast path-attribute discard**]	Displays all prefixes for which an attribute has been discarded.
show bgp {**ipv4** | **ipv6**} **unicast path-attribute unknown**]	Displays all prefixes that have an unknown attribute.
show bgp {**ipv4** | **ipv6**} **unicast** *ip-address*	Displays the unknown attributes and discarded attributes associated with a prefix.

The following example shows the prefixes for which an attribute has been discarded:

```
switch# show bgp ipv4 unicast path-attribute discard
Network            Next Hop
1.1.1.1/32         20.1.1.1
1.1.1.2/32         20.1.1.1
1.1.1.3/32         20.1.1.1
```

The following example shows the prefixes that have an unknown attribute:

```
switch# show bgp ipv4 unicast path-attribute unknown
Network            Next Hop
2.2.2.2/32         20.1.1.1
2.2.2.3/32         20.1.1.1
```

The following example shows the unknown attributes and discarded attributes associated with a prefix:

```
switch# show bgp ipv4 unicast 2.2.2.2
BGP routing table entry for 2.2.2.2/32, version 6241
Paths: (1 available, best #1, table default)
  Not advertised to any peer
  Refresh Epoch 1
```

```
1000
  20.1.1.1 from 20.1.1.1 (20.1.1.1)
    Origin IGP, localpref 100, valid, external, best
    unknown transitive attribute: flag 0xE0 type 0x62 length 0x64
      value 0000 0000 0100 0000 0200 0000 0300 0000
            0400 0000 0500 0000 0600 0000 0700 0000
            0800 0000 0900 0000 0A00 0000 0B00 0000
            0C00 0000 0D00 0000 0E00 0000 0F00 0000
            1000 0000 1100 0000 1200 0000 1300 0000
            1400 0000 1500 0000 1600 0000 1700 0000
            1800 0000
    rx pathid: 0, tx pathid: 0x0
    Updated on Jul 20 2019 07:50:43 PST
```

BGP Graceful Shutdown

About BGP Graceful Shutdown

Beginning with release 9.3(1), BGP supports the graceful shutdown feature. This BGP feature works with the BGP **shutdown** command to:

- Dramatically decrease the network convergence time when a router or link is taken offline.
- Reduce or eliminate dropped packets that are in transit when a router or link is taken offline.

Despite the name, BGP graceful shutdown does not actually cause a shutdown. Instead, it alerts connected routers that a router or link will be going down soon.

The graceful shutdown feature uses the GRACEFUL_SHUTDOWN well-known community (0xFFFF0000 or 65535:0), which is identified by IANA and the IETF through RFC 8326. This well-known community can be attached to any routes, and it is processed like any other attribute of a route.

Because this feature announces that a router or link will be going down, the feature is useful in preparation of maintenance windows or planned outages. Use this feature before shutting down BGP to limit the impact on traffic.

Graceful Shutdown Aware and Activate

BGP routers can control the preference of all routes with the GRACEFUL_SHUTDOWN community through the concept of GRACEFUL SHUTDOWN awareness. Graceful shutdown awareness is enabled by default, which enables the receiving peers to deprefer incoming routes carrying the GRACEFUL_SHUTDOWN community. Although not a typical use case, you can disable and reenable graceful shutdown awareness through the **graceful-shutdown aware** command.

Graceful shutdown aware is applicable only at the BGP global context. For information about contexts, see Graceful Shutdown Contexts, on page 55. The aware option operates with another option, the **activate** option, which you can assign to a route map for more granular control over graceful shutdown routes.

Interaction of the Graceful Shutdown Aware and Activate Options

When a graceful shutdown is activated, the GRACEFUL_SHUTDOWN community is appended to route updates only when you specify the **activate** keyword. At this point, new route updates that contain the

community are generated and transmitted. When the **graceful-shutdown aware** command is configured, all routers that receive the community then deprefer (lower the route preference of) the routes in the update. Without the **graceful-shutdown aware** command, BGP does not deprefer routes with the GRACEFUL_SHUTDOWN community.

After the feature is activated and the routers are aware of graceful shutdown, BGP still considers the routes with the GRACEFUL_SHUTDOWN community as valid. However, those routes are given the lowest priority in the best-path calculation. If alternate paths are available, new best paths are chosen, and convergence occurs to accommodate the router or link that will soon go down.

Graceful Shutdown Contexts

BGP graceful shutdown feature has two contexts that determine what the feature affects and what functionality is available.

Context	Affects	Commands
Global	The entire switch and all routes processed by it. For example, readvertise all routes with the GRACEFUL_SHUTDOWN community.	**graceful-shutdown activate [route-map** *route-map***]** **graceful-shutdown aware**
Peer	A BGP peer or a link between neighbors. For example, advertise only one link between peers with GRACEFUL_SHUTDOWN community.	**graceful-shutdown activate [route-map** *route-map***]**

Graceful Shutdown with Route Maps

Graceful shutdown works with the route policy manager (RPM) feature to control how the switch's BGP router transmits and receives routes with the GRACEFUL_SHUTDOWN community. Route maps can process route updates with the community in the inbound and outbound directions. Typically, route maps are not required. However, if needed, you can use them to customize the control of graceful shutdown routes.

Normal Inbound Route Maps

Normal inbound route maps affect routes that are incoming to the BGP router. Normal inbound route maps are not commonly used with the graceful shutdown feature because routers are aware of graceful shutdown by default.

Cisco Nexus switches running Cisco NX-OS Release 9.3(1) and later do not require an inbound route map for the graceful shutdown feature. Cisco NX-OS Release 9.3(1) and later have implicit inbound route maps that automatically deprefer any routes that have the GRACEFUL_SHUTDOWN community if the BGP router is graceful shutdown aware.

Normal inbound route maps can be configured to match against the well-known GRACEFUL_SHUTDOWN community. Although these inbound route maps are not common, there are some cases where they are used:

- If switches are running a Cisco NX-OS release earlier than 9.3(1), they do not have the implicit inbound route map present in NX-OS 9.3(1). To use the graceful shutdown feature on these switches, you must

create a graceful shutdown inbound route map. The route map must match inbound routes with the well-known GRACEFUL_SHUTDOWN community, permit them, and deprefer them. If an inbound route map is needed, create it on the BGP peer that is running a version of NX-OS earlier than 9.3(1) and is receiving the graceful shutdown routes.

- If you want to disable graceful shutdown aware, but still want the router to act on incoming routes with GRACEFUL_SHUTDOWN community from some BGP neighbors, you can configure an inbound route map under the respective peers.

Normal Outbound Route Maps

Normal outbound route maps control forwarding the routes that a BGP router sends. Normal outbound route maps can affect the graceful shutdown feature. For example, you can configure an outbound route map to match on the GRACEFUL_SHUTDOWN community and set attributes, and it takes precedence over any graceful shutdown outbound route maps.

Graceful Shutdown Outbound Route Maps

Outbound Graceful shutdown route maps are specific type of outbound route map for the graceful shutdown feature. They are optional, but they are useful when you already have a community list that is associated with a route map. The typical graceful shutdown outbound route map contains only `set` clauses to set or modify certain attributes.

You can use outbound route maps in the following ways:

- For customers that already have existing outbound route maps, you can add a new entry with a higher sequence number, match on the GRACEFUL_SHUTDOWN well-known community, and add any attributes that you want.
- You can also use a graceful shutdown outbound route map with the **graceful-shutdown activate route-map** *name* option. This is the typical use case.

  This route map requires no match clauses, so the route map matches on all routes being sent to the neighbor.

Route Map Precedence

When multiple route maps are present on the same router, the following order of precedence is applied to determine how routes with the community are processed:Consider the following example. Assume you have a standard outbound route map name Red that sets a local-preference of 60. Also, assume you have a peer graceful-shutdown route map that is named Blue that sets local-pref to 30. When the route update is processed, the local preference will be set to 60 because Red overwrites Blue.

- Normal outbound route maps take precedence over peer graceful shutdown maps.
- Peer graceful shutdown maps take precedence over global graceful shutdown maps.

Guidelines and Limitations

The following are limitations and guidelines for BGP global shutdown:

- Graceful shutdown feature can only help avoid traffic loss when alternative routes exist in the network for the affected routers. If the router has no alternate routes, routes carrying the

GRACEFUL_SHUTDOWN community are the only ones available, and therefore, are used in the best-path calculation. This situation defeats the purpose of the feature.

- Configuring a BGP send community is required to send the GRACEFUL_SHUTDOWN community.
- For route maps:
  - When global route maps and neighbor route maps are configured, the per-neighbor route maps take precedence.
  - Outbound route maps take precedence over any global route maps configured for graceful shutdown.
  - Outbound route maps take precedence over any peer route maps configured for graceful shutdown.
  - To add the graceful shutdown functionality to legacy (existing) inbound route maps, follow this order:
    1. Add the graceful shutdown match clause to the top of the route map by setting a low sequence number for the clause (for example, sequence number 0).
    2. Add a continue statement after the graceful shutdown clause. If you omit the continue statement, route-map processing stops when it matches the graceful shutdown clause, any other clauses with higher sequence numbers (for example, 1 and higher) are not processed.

Graceful Shutdown Task Overview

To use the graceful shutdown feature, you typically enable graceful-shutdown aware on all Cisco Nexus switches and leave the feature enabled. When a BGP router must be taken offline, you configure graceful-shutdown activate on it.

The following details document the best practice for using the graceful shutdown feature.

To bring the router or link down:

1. Configure the Graceful Shutdown feature.
2. Watch the neighbor for the best path.
3. When the best path is recalculated, issue the **shutdown** command to disable BGP.
4. Perform the work that required you to shut down the router or link.

To bring the router or link back online:

1. When you finish the work that required the shutdown, reenable BGP (**no shutdown**).
2. Disable the graceful shutdown feature (**no graceful-shutdown activate** in config router mode).

Configuring Graceful Shutdown on a Link

This task enables you to configure graceful shutdown on a specific link between two BGP routers.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

	Command or Action	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch-1(config)# router bgp 110`
	`switch-1(config-router)#`
		Enters router configuration mode to create or configure a BGP routing process.

Step 3	**neighbor {** *ipv4-address*|*ipv6-address* **}** **remote-as** *as-number*
	Example:
	`switch-1(config-router)# neighbor 10.0.0.3 remote-as 200`
	`switch-1(config-router-neighbor)#`
		Configures the autonomous system (AS) to which the neighbor belongs.

Step 4	**graceful-shutdown activate [route-map** *map-name***]**
	Example:
	`switch-1(config-router-neighbor)# graceful-shutdown activate route-map gshutPeer`
	`switch-1(config-router-neighbor)#`
		Configures graceful shutdown on the link to the neighbor. Also, advertises the routes with the well-known GRACEFUL_SHUTDOWN community and applies the route map to the outbound route updates.
		The routes are advertised with the graceful-shutdown community by default. In this example, routes are advertised to the neighbor with the Graceful-shutdown community with a route-map named gshutPeer.
		The devices receiving the gshut community look at the communities of the route and optionally use the communities to apply routing policy.

Filtering BGP Routes and Setting Local Preference Based On GRACEFUL_SHUTDOWN Communities

Switches that are not yet running 9.3(1) do not have an inbound route map that matches against the GRACEFUL_SHUTDOWN community name. Therefore, they have no way of identifying and depreferring the correct routes.

For switches running a release of NX-OS that is earlier than 9.3(1), you must configure an inbound route map that matches on the community value for graceful shutdown (65535:0) and deprefers routes.

If your switch is running 9.3(1) or later, you do not need to configure an inbound route map.

Procedure

	Command or Action	Purpose

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

Step 2	**ip community list standard** *community-list-name* **seq** *sequence-number* **{ permit | deny }** *value*
	Example:
	`switch-1(config)# ip community-list standard GSHUT seq 10 permit 65535:0`
	`switch-1(config)#`
		Configures a community list and permits or denies routes that have the well-known graceful shutdown community value.

Step 3	**route map** *map-tag* **{deny | permit}** *sequence-number*
	Example:
	`switch-1(config)# route-map RM_GSHUT permit 10`
	`switch-1(config-route-map)#`
		Configures a route map as sequence 10 and permits routes that have the GRACEFUL_SHUTDOWN community.

Step 4	**match community** *community-list-name*
	Example:
	`switch-1(config-route-map)# match community GSHUT`
	`switch-1(config-route-map)#`
		Configures that routes that match the IP community list GSHUT are processed by Route Policy Manager (RPM).

Step 5	**set local-preference** *local-pref-value*
	Example:
	`switch-1(config-route-map)# set local-preference 10`
	`switch-1(config-route-map)#`
		Configures that the routes that match the IP community list GSHUT will be given a specified local preference.

Step 6	**exit**
	Example:
	`switch-1(config-route-map)# exit`
	`switch-1(config)#`
		Leaves route map configuration and returns to global configuration mode.

Step 7	**router bgp** *community-list-name*
	Example:
	`switch-1(config)# router bgp 100`
	`switch-1(config-router)#`
		Enters router configuration mode and creates a BGP instance.

Step 8	**neighbor {** *ipv4-address|ipv6-address* **}**
	Example:
	`switch-1(config-router)# neighbor 10.0.0.3`
	`switch-1(config-router-neighbor)#`
		Enters route BGP neighbor mode for a specified neighbor.

	Command or Action	Purpose
Step 9	**address-family {** *address-family sub family* **}** **Example:** `nxosv2(config-router-neighbor)# address-family ipv4 unicast nxosv2(config-router-neighbor-af)#`	Puts the neighbor into address family (AF) configuration mode.
Step 10	**send community** **Example:** `nxosv2(config-router-neighbor-af)# send-community nxosv2(config-router-neighbor-af)#`	Enables BGP community exchange with the neighbor.
Step 11	**route map** *map-tag* **in** **Example:** `nxosv2(config-router-neighbor-af)# route-map RM_GSHUT in nxosv2(config-router-neighbor-af)#`	Applies the route map to incoming routes from the neighbor. In this example, the route map that is named RM_GSHUT permits routes with the GRACEFUL_SHUTDOWN community from the neighbor.

Configuring Graceful Shutdown for All BGP Neighbors

You can manually apply the GRACEFUL_SHUTDOWN well-known community to all the neighbors of a graceful shutdown initiator.

You can configure graceful shutdown at the global level for all BGP neighbors.

Before you begin

If you have not already enabled BGP, enable it now (**feature bgp**).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch-1# configure terminal switch-1(config)#`	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system-number* **Example:** `switch-1(config)# router bgp 110 switch-1(config-router)#`	Enters router configuration mode to create or configure a BGP routing process.
Step 3	**graceful-shutdown activate [route-map** *map-name***]** **Example:** `switch-1(config-router-neighbor)# graceful-shutdown activate route-map`	Configures graceful shutdown route map for the links to all neighbors. Also, advertises all routes with the well-known GRACEFUL_SHUTDOWN community and applies the route map to the outbound route updates.

	Command or Action	Purpose
	gshutPeer switch-1(config-router-neighbor)#	The routes are advertised with the GRACEFUL_SHUTDOWN community by default. In this example, routes are advertised to all neighbors with the community with a route-map named gshutPeer. The route map should contain only set clauses. The devices receiving the GRACEFUL_SHUTDOWN community look at the communities of the route and optionally use the communities to apply routing policy.

Controlling the Preference for All Routes with the GRACEFUL_SHUTDOWN Community

Cisco NX-OS enables lowering the preference of incoming routes that have the GRACEFUL_SHUTDOWN community. When **graceful shutdown aware** is enabled, BGP considers routes carrying the community as the lowest preference during best path calculation. By default, lowering the preference is enabled, but you can selectively disable this option.

Whenever you enable or disable this option, you trigger a BGP best-path calculation. This option gives you the flexibility to control the behavior of the BGP best-path calculation for the graceful shutdown well-known community.

Before you begin

If you have not enabled BGP, enable it now (**feature bgp**).

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** switch-1(config)# **config terminal** switch-1(config)#	Enters global configuration mode.
Step 2	**router bgp** *autonoums-system* **Example:** switch-1(config)# **router bgp 100** switch-1(config-router)#	Enters router configuration mode and configures a BGP routing process.
Step 3	(Optional) **no graceful-shutdown aware** **Example:** switch-1(config-router)# **no graceful-shutdown aware** switch-1(config-router)#	For this BGP router, do not give lower preference for all routes that have the GRACEFUL_SHUTDOWN community. The default action is to deprefer routes when the graceful shutdown aware feature is disabled, so using the **no** form of the command is optional for not depreferring graceful shutdown routes.

Preventing Sending the GRACEFUL_SHUTDOWN Community to a Peer

If you no longer need the GRACEFUL_SHUTDOWN community that is appended as a route attribute to outbound route updates, you can remove the community, which no longer sends it to a specified neighbor. One use case would be when a router is at an autonomous system boundary, and you do not want the graceful shutdown functionality to propagate outside of an autonomous system boundary.

To prevent sending the GRACEFUL_SHUTDOWN to a peer, you can disable the send community option or strip the community from the outbound route map.

Choose either of the following methods:

- Disable the send-community in the running config.

  Example:

  ```
  nxosv2(config-router-neighbor-af)# no send-community standard
  nxosv2(config-router-neighbor-af)#
  ```

  If you use this option, the GRACEFUL_SHUTDOWN community is still received by the switch, but it is not sent to the downstream neighbor through the outbound route map. All standard communities are not sent either.

- Delete the GRACEFUL_SHUTDOWN community through an outbound route map by following these steps:

  1. Create an IP community list matches the GRACEFUL_SHUTDOWN community.
  2. Create an outbound route map to match against the GRACEFUL_SHUTDOWN community.
  3. Use a **set community-list delete** clause to strip GRACEFUL_SHUTDOWN community.

  If you use this option, the community list matches and permits the GRACEFUL_SHUTDOWN community, then the outbound route map matches against the community and then deletes it from the outbound route map. All other communities pass through the outbound route map without issue.

Displaying Graceful Shutdown Information

Information about the graceful shutdown feature is available through the following **show** commands.

Command	Action
show ip bgp community-list graceful-shutdown	Shows all entires in the BGP routing table that have the GRACEFUL_SHUTDOWN community.
show running-config bgp	Shows the running BGP configuration.
show running-config bgp all	Shows all information for the running BGP configuration including information about the graceful shutdown feature.

Command	Action
show bgp *address-family* **neighbors** *neighbor-address*	When the feature is configured for the peer, shows the following: • The state of the graceful-shutdown-activate feature for the specified neighbor • The name of any graceful shutdown route map configured for the specified neighbor
show bgp process	Shows different information depending on the context. When the graceful-shutdown-activate option is configured in peer context, shows the enabled or disabled state for the feature through `graceful-shutdown-active`. When the graceful-shutdown-activate option is configured in global context and has a graceful-shutdown route map, shows the enabled state of the feature through the following: • `graceful-shutdown-active` • `graceful-shutdown-aware` • `graceful-shutdown route-map`
show ip bgp *address*	For the specified address, shows the BGP routing table information, including the following: • The state of the specified address as the best path • Whether the specified address is part of the GRACEFUL_SHUTDOWN community

Graceful Shutdown Configuration Examples

These examples show some configurations for using the graceful shutdown feature.

Configuring Graceful Shutdown for a BGP Link

The following example shows how to configure graceful shutdown while setting a local preference and a community:

- Configuring graceful shutdown activate for the link to the specified neighbor
- Adding the GRACEFUL_SHUTDOWN community to the routes
- Setting a route map named gshutPeer with only set clauses for outbound routes with the community.

```
router bgp 100
    neighbor 20.0.0.3 remote-as 200
         graceful-shutdown activate route-map gshutPeer
         address-family ipv4 unicast
```

```
                send-community

 route-map gshutPeer permit 10
     set local-preference 0
     set community 200:30
```

Configuring Graceful Shutdown for All-Neighbor BGP Links

The following example shows:

- Configuring graceful shutdown activate for all the links connecting the local router and all its neighbors.
- Adding the GRACEFUL_SHUTDOWN community to the routes.
- Setting a route map that is named gshutAall with only set clauses for all outbound routes.

```
router bgp 200
   graceful-shutdown activate route-map gshutAll

route-map gshutAll permit 10
   set as-path prepend 10 100 110
   set community 100:80

route-map Red permit 10
   set local-pref 20

router bgp 100
   graceful-shutdown activate route-map gshutAll
      router-id 2.2.2.2
         address-family ipv4 unicast
         network 2.2.2.2/32
         neighbor 1.1.1.1 remote-as 100
         update-source loopback0
         address-family ipv4 unicast
            send-community
         neighbor 20.0.0.3 remote-as 200
         address-family ipv4 unicast
            send-community
               route-map Red out
```

In this example, the `gshutAll` route-map takes effect for neighbor 1.1.1.1, but not neighbor 20.0.0.3, because the outbound route-map `Red` configured under neighbor 20.0.0.3 takes precedence instead.

Configuring Graceful Shutdown Under a Peer-Template

This example configures the graceful shutdown feature under a peer-session template, which is inherited by a neighbor.

```
router bgp 200
   template peer-session p1
      graceful-shutdown activate route-map gshut_out
   neighbor 1.1.1.1 remote-as 100
      inherit peer-session p1
      address-family ipv4 unicast
         send-community
```

Filtering BGP Routes and Setting Local Preference Based on GRACEFUL_SHUTDOWN Community Using and Inbound Route Map

This example shows how to use a community list to filter the incoming routes that have the GRACEFUL_SHUTDOWN community. This configuration is useful for legacy switches that are not running Cisco NX-OS 9.3(1) as a minimum version.

The following example shows:

- An IP Community List that permits routes that have the GRACEFUL_SHUTDOWN community.
- A route map that is named RM_GSHUT that permits routes based on a standard community list named GSHUT.
- The route map also sets the preference for the routes it processes to 0 so that those routes are given lower preference for best path calculation when the router goes offline. The route map is applied to incoming IPv4 routes from the neighbor (20.0.0.2).

```
ip community-list standard GSHUT permit 65535:0

route-map RM_GSHUT permit 10
   match community GSHUT
   set local-preference 0

router bgp 200
   neighbor 20.0.0.2 remote-as 100
      address-family ipv4 unicast
         send-community
         route-map RM_GSHUT in
```

Verifying the Advanced BGP Configuration

To display the BGP configuration information, 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 ip {**unicast**} [*ip-address*] **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 [**vrf** *vrf-name*] **ip** {**unicast**} [*ip-address*] **community-list** list-name [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp ip {**unicast**} [*ip-address*] **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 ip {**unicast**} [ip-address] **extcommunity-list** *list-name* [**exact-match**] [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community list.

Command	Purpose
show bgp ip {**unicast**} [*ip-address*] {**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 ip {**unicast**} [*ip-address*] **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp ip {**unicast**} [*ip-address*] **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.
show bgp ip {**unicast**} [*ip-address*] **neighbors** [*ip-address*] [**vrf** *vrf-name*]	Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors.
show bgp ip {**unicast**} [*ip-address*] {**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 ip {**unicast**} [*ip-address*] **policy name** [**vrf** *vrf-name*]	Displays the BGP policy information. Use the **clear bgp policy** command to clear the policy information.
show bgp ip {**unicast**} [*ip-address*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp ip {**unicast**} [*ip-address*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp ip {**unicast**} [*ip-address*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp ip {**unicast**} [*ip-address*] **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 ip bgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the Cisco Nexus 3000 Series Command Reference for more information.
show ip mbgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the Cisco Nexus 3000 Series Command Reference for more information.

Command	Purpose
show running-configuration bgp	Displays the current running BGP configuration.

Displaying BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
show bgp ip {**unicast**} [*ip-address*] **flap-statistics** [**vrf** *vrf-name*]	Displays the BGP route flap statistics. Use the **clear bgp flap-statistics** command to clear these statistics.
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics.
show bgp sessions [**vrf** *vrf-name*]	Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics.
show bgp statistics	Displays the BGP statistics.

Configuration Examples for BFD for BGP

This example shows how to enable BFD for individual BGP neighbors:

```
router bgp 400
  router-id 2.2.2.2
  neighbor 172.16.2.3
     bfd
     remote-as 400
     update-source Vlan1002
     address-family ipv4 unicast
```

This example shows how to enable BFD for BGP prefix peers:

```
router bgp 400
  router-id 1.1.1.1
  neighbor 172.16.2.0/24
    bfd
    remote-as 400
    update-source Vlan1002
    address-family ipv4 unicast
```

Related Topics

The following topics can give more information on BGP:

- Configuring Advanced BGP, on page 1
- Configuring Route Policy Manager

Additional References

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

Related Documents

Related Topic	Document Title
BGP CLI commands	Cisco Nexus 3000 Series Command Reference

MIBs

MIBs	MIBs Link
BGP4-MIB CISCO-BGP4-MIB	To locate and download MIBs, go to the following: MIB Locator.