



Configuring Advanced BGP

This chapter contains the following sections:

- [About Advanced BGP, on page 2](#)
- [Prerequisites for Advanced BGP, on page 14](#)
- [Guidelines and Limitations for Advanced BGP, on page 14](#)
- [Default Settings, on page 19](#)
- [Configuring Advanced BGP, on page 19](#)
- [Configuring BGP Additional Paths, on page 38](#)
- [Configuring eBGP, on page 42](#)
- [Configuring AS Confederations, on page 46](#)
- [Configuring Route Reflector, on page 47](#)
- [Configuring Next-Hops on Reflected Routes Using an Outbound Route-Map, on page 49](#)
- [Configuring Route Dampening, on page 52](#)
- [Configuring Load Sharing and ECMP, on page 52](#)
- [Unequal Cost Multipath \(UCMP\) over BGP, on page 53](#)
- [Enabling UCMP over BGP, on page 53](#)
- [Guidelines and Limitations for UCMP over BGP, on page 53](#)
- [Configuring Maximum Prefixes, on page 53](#)
- [Configuring DSCP, on page 54](#)
- [Configuring Dynamic Capability, on page 55](#)
- [Configuring Aggregate Addresses, on page 55](#)
- [Suppressing BGP Routes, on page 56](#)
- [Configuring BGP Conditional Advertisement, on page 57](#)
- [Configuring Route Redistribution, on page 59](#)
- [DMZ Link Bandwidth, on page 60](#)
- [Advertising the Default Route, on page 72](#)
- [Configuring BGP Attribute Filtering and Error Handling, on page 73](#)
- [Tuning BGP, on page 76](#)
- [Configuring Policy-Based Administrative Distance, on page 80](#)
- [Configuring Multiprotocol BGP, on page 82](#)
- [Configuring BMP, on page 83](#)
- [BGP Local Route Leaking, on page 86](#)
- [BGP Graceful Shutdown, on page 93](#)
- [Configuring a Graceful Restart, on page 105](#)

- [Configuring Virtualization, on page 108](#)
- [Verifying the Advanced BGP Configuration, on page 109](#)
- [Monitoring BGP Statistics, on page 111](#)
- [Configuration Examples, on page 112](#)
- [Related Topics, on page 112](#)
- [Additional References, on page 113](#)

About Advanced BGP

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

Peer Templates

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

Cisco NX-OS implements three types of peer templates:

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

Authentication

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



Note The MD5 password must be identical between BGP peers.

Route Policies and Resetting BGP Sessions

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

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

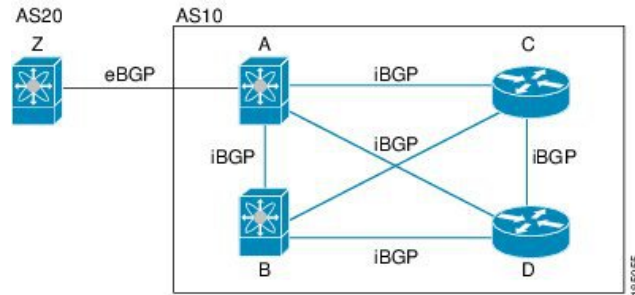
Typically eBGP peerings need to be over directly connected interfaces so that convergence will be faster when the interface goes down.

iBGP

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

The 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.

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

You should use loopback interfaces for establishing iBGP 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, on page 42](#) section for information on multihop, fast external fallovers, and limiting the size of the AS_path attribute.



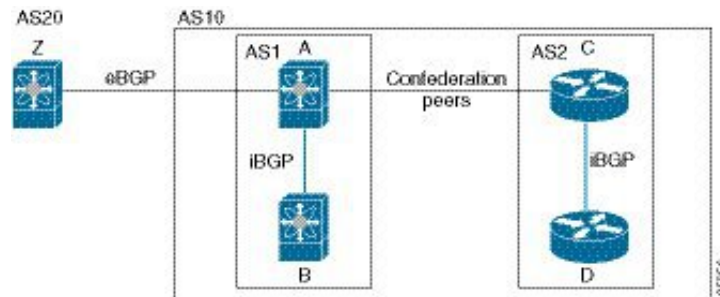
Note You should configure a separate interior gateway protocol in the iBGP network.

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 figure shows the BGP network, 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.

Route Reflector

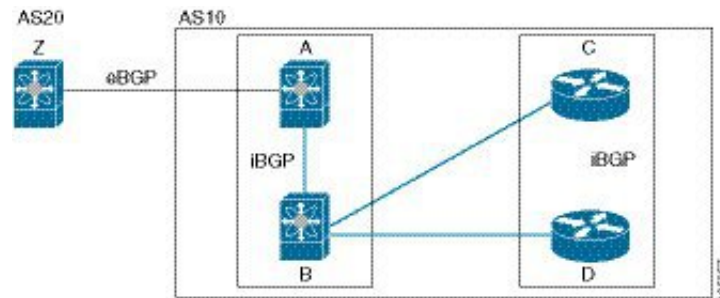
You can alternately reduce the iBGP mesh by using a route reflector configuration where route reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

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.

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

In the 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 that are supported by a peer by using the capabilities negotiation feature. Capabilities negotiation allows BGP to use only the set of features supported by both BGP peers on a link.

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

Route Dampening

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

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

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

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



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

Load Sharing and Multipath

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

To configure as-path multipath-relax command effectively, configure the command per VRF under BGP. Also, configure as-path multipath-relax command under the custom VRF so that multiple routers get installed in the custom VRF Route-Target (RT).

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. For more information, see the [BGP Additional Paths](#) section.



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



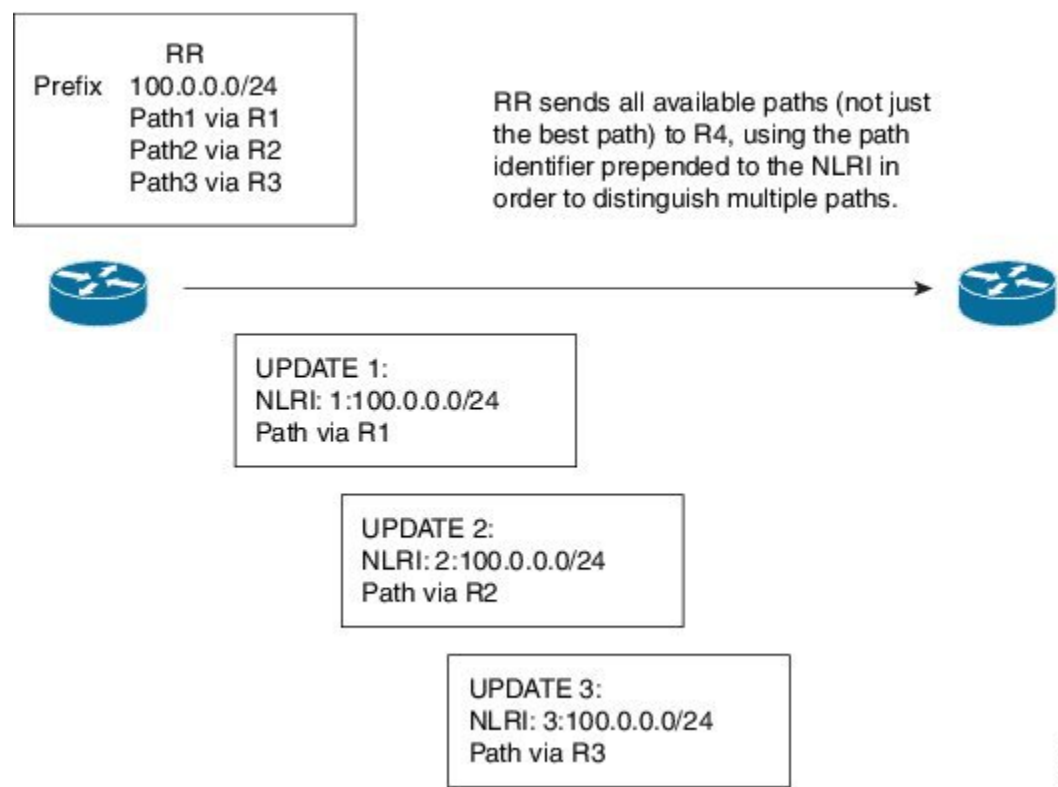
Note When you configure a route reflector for iBGP multipath, and the route reflector advertises the selected best path to its peers, the next hop for the path is not modified.

BGP Additional Paths

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

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

Figure 4: BGP Route Advertisement with the Additional Paths Capability



For information on configuring BGP additional paths, see the [Configuring BGP Additional Paths, on page 38](#) section.

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.

A summary entry is created in the BGP table when you issue the **aggregate-address** command, but the summary entry is not eligible for advertisement until a subset of the aggregate is found in the table.

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 Routing Information Base (RIB) that may affect BGP next-hop reachability.

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

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



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

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

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

See the [Configuring BGP Next-Hop Address Tracking](#) section for more information.

Route Redistribution

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

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

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

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

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

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

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) for IPv4 and IPv6. BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection

between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules.

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

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.

Beginning with Cisco NX-OS Release 9.3(3), BFD supports BGP Interface Peering via IPv6 Link-Local for IPv4 and IPv6 Address Families. However, BFD multihop is not supported with unnumbered BGP.

See the [Cisco Nexus 9000 Series NX-OS Interfaces Configuration Guide](#) for more information.

Tuning BGP

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

BGP Timers

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

Tuning the Best-Path Algorithm

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

Multiprotocol BGP

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

RFC 5549

BGP supports RFC 5549, which allows an IPv4 prefix to be carried over an IPv6 next hop. Because BGP is running on every hop, all routers can forward IPv4 and IPv6 traffic. Therefore, 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).

Beginning with Cisco NX-OS Release 9.2(2), Cisco Nexus 9500 platform switches with -R line cards support RFC 5549.

Currently, NX-OS does not support IPv6 recursive next-hops (RNH) for an IPv4 route.

RFC 6368

Introduction

This section describes how the Internal Border Gateway Protocol (iBGP) between Provider Edge (PE) and Customer Edge (CE) feature is implemented in Cisco NX-OS.

In current deployments, when BGP is used as the Provider/Customer Edge routing protocol, these peering sessions are configured as an external peering between the VPN provider autonomous system (AS) and the customer network autonomous system.

RFC 6368 adds support for these peers to be configured as iBGP peers instead.

Beginning with Cisco NX-OS Release 10.1(2), RFC 6368 support is enabled for EVPN-VxLANv4 and EVPN-VxLANv6.

Framework

Beginning with Cisco NX-OS Release 10.1(2), deploying iBGP PE-CE feature:

- You can have one single Autonomous System Number (ASN) on the multiple sites of the VRF, without the deployment of External Border Gateway Protocol (eBGP) with as-override.
- You can give internal route reflection towards the CE routers, acting as if the Provider core is one transparent Route Reflector (RR).

With this feature, the VRF sites can have the same ASN as the provider core. However, in case the ASN of the VRF sites are different than the ASN of the provider core, it can be made to appear the same with the use of the feature local Autonomous System (AS).

Implement iBGP PE-CE

Here are the two major parts to make this feature work:

- A new attribute `ATTR_SET` added to the BGP protocol to carry the VPN BGP attributes across the provider core in a transparent manner.
- Make the PE router a RR for the iBGP sessions towards the CE routers in the VRF.

The new `ATTR_SET` attribute allows the provider to carry all the BGP attributes of the customer transparently and does not interfere with the provider attributes and BGP policies. Such attributes are the cluster list, local preference, and so on.

BGP Customer Route Attribute

`ATTR_SET` is the new BGP attribute used to carry the VPN BGP attributes of the provider customer. It is an optional transitive attribute. In this attribute, Local Preference, Med, Origin, AS Path, Originator ID, Cluster list attributes will be carried across the provider network. The `ATTR_SET` attribute has the format:

```
+-----+
| Attr Flags (O|T) Code = 128 |
+-----+
| Attr. Length (1 or 2 octets) |
+-----+
| Origin AS (4 octets)      |
+-----+
|Path Attributes (variable) |
+-----+
```

- Attribute Flags are regular BGP attribute flags.
- Attribute length indicates whether the length is one or two octets.
- Origin AS field is to prevent a leak of one route that originated in one AS to be leaked to another AS without proper manipulation of the `AS_PATH`.
- The variable-length path attributes field carries VPN BGP attributes that must be carried across the provider core.

For more information on the implementation of iBGP PE-CE, see [IOS Implementation of the iBGP PE-CE Feature](#).

This example shows BGP neighbor configuration on PE device for iBGP Customer Edge device:

```
router bgp 200
vrf nxbgp3-leaf2-2
address-family ipv4 unicast
redistribute static route-map ALLOW-ALL
address-family ipv6 unicast
redistribute static route-map ALLOW-ALL
neighbor 101.101.101.101 remote-as 200
description ibgp sample config
internal-vpn-client (1)
address-family ipv4 unicast
route-reflector-client (2)
next-hop-self (3)
```

BGP Monitoring Protocol

The BGP Monitoring Protocol (BMP) monitors BGP updates and peer statistics and is supported for all Cisco Nexus 9000 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.

Graceful Restart and High Availability

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

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

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

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

When a router detects that a graceful restart operation is in progress, both routers exchange their topology tables. When the router has route updates from all BGP peers, it removes all the stale routes and runs the best-path algorithm on the updated routes.

After the switchover, Cisco NX-OS applies the running configuration, and BGP informs the neighbors that it is operational again.

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

Beginning with Cisco NX-OS Release 9.3(3), BGP prefix peers support graceful restarts.

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

Low Memory Handling

BGP reacts to low memory for the following conditions:

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



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

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

See the [Tuning BGP](#) section for more information on how to exempt a BGP peer from a shutdown due to a low memory condition.

Virtualization Support

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

Prerequisites for Advanced BGP

Advanced BGP has the following prerequisites:

- You must enable BGP (see the [Enabling BGP](#) 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

Advanced BGP has the following configuration guidelines and limitations:

- There are three scenarios in which the command behavior has changed beginning with Cisco NX-OS Release 9.3(5):

```
• Router bgp 1
  Template peer abc
    Ttl-security hops 30
  Neighbor 1.2.3.4
    Inherit peer abc
```

If you later enter the **ebgp-multihop 20** command, the configuration is blocked due to the presence of **ttl-security hops 30** command. Beginning with the Cisco NX-OS Release 9.3(5), the configuration is no longer blocked. However, the **ttl-security hops** command has priority and would be the enabled functionality.

```
• Router bgp 1
  Template peer abc
    Ebgp-multihops 20
  Neighbor 1.2.3.4
    Inherit peer abc
```

If you later enter the **ttl-security hops 30** command, the configuration is blocked due to the presence of **ebgp-multihop 20** command. Beginning with Cisco NX-OS Release 9.3(5), the configuration

is no longer blocked. However again, the **ttl-security hops** command has priority and would be the enabled functionality.

- Router bgp 1
 Template peer abc
 Remote-as 1
 Neighbor 1.2.3.4
 Inherit peer abc

If you later enter the **ttl-security hops 30** or **ebgp-multihop 20** commands, they are blocked. Beginning with Cisco NX-OS Release 9.3(5), the configuration is not blocked. However, their functionalities are turned off as the **remote-as** command has priority which makes the peer an iBGP peer.

- Prefix peering operates only in passive TCP mode. It accepts incoming connections from remote peers if the peer address falls within the prefix.
- Beginning with Cisco NX-OS 9.3(5), a packet with a TTL value of 1 to a vPC peer is hardware that is forwarded.
- Configuring the **advertise-maps** command multiple times is not supported.
- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying uppercase and lowercase characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are not two different entries.
- The dynamic AS number prefix peer configuration overrides the individual AS number configuration that is 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 that are 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 that are received and system resources used.
- Configure the update source to establish a session with eBGP multihop sessions.
- Specify a BGP route map if you configure a redistribution.
- Configure the BGP router ID within a VRF.
- If you decrease the keepalive and hold timer values, the network might experience session flaps.
- When you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an extra deny statement into the route map.
- To enable BFD for iBGP single-hop peers, you must configure the **update-source** option on the physical interface.
- Beginning with Cisco NX-OS Release 9.3(3), BFD for BGP is supported for BGP IPv4 and IPv6 prefix peers.
- The following guidelines and limitations apply to the **remove-private-as** command:

- It applies only to eBGP peers.
 - It applies only to routers in a public AS only. The workaround to this restriction would be to apply the **neighbor local-as** command on a per-neighbor basis, with the local AS number being a public AS number.
 - It can be configured only in neighbor configuration mode and not in neighbor-address-family mode.
 - If the AS-path includes both private and public AS numbers, the private AS numbers are not removed.
 - If the AS-path contains the AS number of the eBGP neighbor, the private AS numbers are not removed.
 - Private AS numbers are removed only if all AS numbers in that AS-path belong to a private AS number range. Private AS numbers are not removed if a peer's AS number or a non-private AS number is found in the AS-path segment.
- If you use the **aggregate-address** command to configure aggregate addresses and the **suppress-fib-pending** command to suppress BGP routes, lossless traffic for aggregates cannot be ensured on BGP or system triggers.
 - When you enable FIB suppression on the switch and route programming fails in the hardware, BGP advertises routes that are not programmed locally in the hardware.
 - If you disable a command in the neighbor, template peer, template peer-session, or template peer-policy configuration mode (and the **inherit peer** or **inherit peer-session** command is present), you must use the **default** keyword to return the command to its default state. For example, to disable the **update-source loopback 0** command from the running configuration, you must enter the **default update-source loopback 0** command.
 - When next-hop-self is configured for route-reflector clients, the route reflector advertises routes to its clients with itself as the next hop.
 - The following guidelines and limitations apply to weighted ECMP:
 - Weighted ECMP is supported only for the IPv4 address family.
 - BGP uses the Link Bandwidth EXTCOMM defined in the draft-ietf-idr-link-bandwidth-06.txt to implement the weighted ECMP feature.
 - BGP accepts the Link Bandwidth EXTCOMM from both iBGP and eBGP peers.
 - 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
```

- When configuring the BGP neighbor with link-local, you need to customize the TCAM "ing-sup" from 512 to 768 except for Cisco Nexus 34XX-S platform, where default carving is sufficient.
- The command [**maximum-paths eibgp**] is supported only in MPLS environments.
- Route-map deletion feature adds a mechanism to block the deletion of entire route-map that is associated with the BGP. With the route-map deletion blocked, the modifications to the route-map statement are still allowed.
- If there are more than one sequence in the route-map, user can still delete any route map sequence until there is at least one sequence available.
- Users can have the forward reference case for route-map from client. However, once route-map is created and associated, the deletion of route-map is blocked.
- Blocking deletion functionality is configurable dynamically using the knob.
- It is allowed to delete the BGP association to the route-map and deletion of route-map itself in a single transaction payload.
- It is allowed to add the BGP association to the route-map and an error must be thrown for deletion of route-map.
- The following is the list of the dual stage related behaviors:
 - If knob and deletion occur together, dual stage has to verify and throw an error without commit.
 - If knob already exists and route-map deletion occurs in dual stage, it must throw an error.
 - If route-map and CLI knob is single commit with different order, it must throw an error.
 - If knob is not enabled and route-map deletion occurs in dual stage, it has to execute successfully.

- In a single verify, if "cli knob is disabled AND route-map deletion" is executed, the route-map deletion is allowed.
- If the route-map used by BGP template is not inherited by any of the BGP neighbors, the entire route-map deletion will still be blocked.
- There are few commands under vrf context that are owned by BGP, but are not part of bgpInst.
- Cloudscale IPv6 link-local BGP support requires carving > 512 ing-sup TCAM region (this requires a reload to take effect).
- As the VPN address family (L3VPN and EVPN) is not supported, the routes received from confederate peers are not advertised in the VPN address family.
- Beginning with Cisco NX-OS Release 10.3(1)F, BGP is supported on the Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, BGP is supported on the Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.3(1)F, VXLAN EVPN is supported only as transit on Cisco Nexus 9808 switches.
- Beginning with Cisco NX-OS Release 10.4(1)F, VXLAN EVPN is supported only as transit on Cisco Nexus 9804 switches.
- Beginning with Cisco NX-OS Release 10.3(3)F, Type-6 encryption for BGP password is supported on Cisco NX-OS switches with the following limitations:
 - If Type-6 encryption is configured, you won't be able to modify the existing Type-6 encrypted password to Type-0/Type-3/Type-7 password.
 - If you downgrade the system by cold reboot with an old image where Type-6 encryption is not supported, ensure to remove the Type-6 configuration and then proceed with cold reboot. Otherwise there will be configuration loss, the results are such that there is no configuration for the neighbor.
 - Primary key configuration is local to the switch. If you take the Type-6 configured running data from one switch and try to apply it on other switch where different primary key is configured, decryption on the new switch will fail.
 - During ISSU, if you migrate from old image (where Type-0/Type-3/Type-7 encrypted keys are there in the configuration) to new image (where Type-6 encryption is supported), BGP won't convert the existing keys to Type-6 encrypted one until or unless reencryption is enforced using the **encryption re-encrypt obfuscated** command.
 - BGP Type-6 passwords will not be supported in non-DME platforms.
 - It is highly recommended for user to specify the password type and password when programmatically (RESTCONF, NETCONF and so on) configuring a neighbor or template's password. When either one of the property is missing in the programmatic call, BGP will use already available (or default) value of the missing property to configure the neighbor or template's password.

If the user has to configure with a property missing then the user has to follow the same sequence of steps in both peer routers.
- Beginning with Cisco NX-OS Release 10.4(1)F, BGP is supported on Cisco Nexus X98900CD-A and X9836DM-A line cards with 9808 and 9804 switches.

Default Settings

The table lists the default settings for advanced BGP parameters.

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

Configuring Advanced BGP

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.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **ip forward**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	interface <i>type slot/port</i> Example: <pre>switch(config)# interface ethernet 1/2 switch(config-if)#</pre>	Enters interface configuration mode.
Step 3	ip forward Example: <pre>switch(config-if)# ip forward</pre>	Allows IPv4 traffic on the interface even when there is no IP address configuration on that interface.

	Command or Action	Purpose
Step 4	(Optional) copy running-config startup-config Example: <pre>switch(config-if)# copy running-config startup-config</pre>	Saves this configuration change.

Configuring BGP Session Templates

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

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

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

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

Before you begin

You must enable BGP (see the [Enabling BGP](#) section).



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.
- When using BGP Peer Template, there is no check for the commands used inside template to verify if that command applies to iBGP/eBGP peer or not. For example if you create a template and add a command "Remove-private-as" inside a template and then assign this template to iBGP peer, then no error will be printed saying this command "Remove-private-as" does not apply to iBGP peer.

SUMMARY STEPS

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

11. (Optional) copy running-config startup-config

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	router bgp <i>autonomous-system-number</i> Example: <pre>switch(config)# router bgp 65535 switch(config-router)#</pre>	Enables BGP and assigns the autonomous system number to the local BGP speaker.
Step 3	template peer-session <i>template-name</i> Example: <pre>switch(config-router)# template peer-session BaseSession switch(config-router-stmp)#</pre>	Enters peer-session template configuration mode.
Step 4	(Optional) password <i>number password</i> Example: <pre>switch(config-router-stmp)# password 0 test</pre>	Adds the clear text password test to the neighbor. The password is stored and displayed in type 3 encrypted form (3DES).
Step 5	(Optional) timers <i>keepalive hold</i> Example: <pre>switch(config-router-stmp)# timers 30 90</pre>	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: <pre>switch(config-router-stmp)# exit switch(config-router)#</pre>	Exits peer-session template configuration mode.
Step 7	neighbor <i>ip-address</i> remote-as <i>as-number</i> Example: <pre>switch(config-router)# neighbor 192.168.1.2 remote-as 65535 switch(config-router-neighbor)#</pre>	Places the router in the neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 8	inherit peer-session <i>template-name</i> Example: <pre>switch(config-router-neighbor)# inherit peer-session BaseSession switch(config-router-neighbor)#</pre>	Applies a peer-session template to the peer.

	Command or Action	Purpose
Step 9	(Optional) description <i>text</i> 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 <i>template-name</i> 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. Use the show bgp neighbor command to see the template applied.

Example

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-af)# 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.



Note Use the **show bgp neighbor** command to see the template applied. See the *Cisco Nexus 9000 Series NX-OS Unicast Routing Command Reference*, for details on all commands available in the template.

Before you begin

You must enable BGP (see the [Enabling BGP](#) 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.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <code>switch# configure terminal</code>	Enters configuration mode.
Step 2	router bgp <i>autonomous-system-number</i> Example: <code>switch(config)# router bgp 65535</code> <code>switch(config-router)#</code>	Enables BGP and assigns the autonomous system number to the local BGP speaker.
Step 3	template peer-session <i>template-name</i> Example: <code>switch(config-router)# template</code> <code>peer-policy BasePolicy</code> <code>switch(config-router-ptmp)#</code>	Creates a peer-policy template.
Step 4	(Optional) advertise-active-only Example: <code>switch(config-router-ptmp)#</code> <code>advertise-active-only</code>	Advertises only active routes to the peer.

	Command or Action	Purpose
Step 5	(Optional) maximum-prefix <i>number</i> Example: switch(config-router-ptmp) # maximum-prefix 20	Sets the maximum number of prefixes allowed from this peer.
Step 6	exit Example: switch(config-router-ptmp) # exit switch(config-router) #	Exits peer-policy template configuration mode.
Step 7	neighbor <i>ip-address remote-as as-number</i> Example: switch(config-router) # neighbor 192.168.1.2 remote-as 65535 switch(config-router-neighbor) #	Places the router in the neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 8	address-family { <i>ipv4 ipv6</i> } { <i>multicast unicast</i> } 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 9	inherit peer-policy <i>template-name preference</i> 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 <i>template-name</i> 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. Use the show bgp neighbor command to see the template applied.

Example

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

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

You must enable BGP (see the [Enabling BGP](#) 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.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*
3. **template peer** *template-name*
4. (Optional) **inherit peer-session** *template-name*
5. (Optional) **address-family** {*ipv4|ipv6*} {*multicast|unicast*}
6. (Optional) **inherit peer-policy** *template-name*
7. **exit**
8. (Optional) **timers** *keepalive hold*
9. **exit**
10. **neighbor** *ip-address* **remote-as** *as-number*
11. **inherit peer** *template-name*
12. (Optional) **timers** *keepalive hold*
13. (Optional) **show bgp peer-template** *template-name*
14. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal	Enters global configuration mode.

	Command or Action	Purpose
Step 2	router bgp <i>autonomous-system-number</i> Example: switch(config)# router bgp 65535	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	template peer <i>template-name</i> Example: switch(config-router)# template peer BasePeer	Enters peer template configuration mode.
Step 4	(Optional) inherit peer-session <i>template-name</i> Example: switch(config-router-neighbor)# inherit peer-session BaseSession	Adds a peer-session template to the peer template.
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-policy <i>template-name</i> Example: switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1	Applies a peer-policy template to the neighbor address family configuration.
Step 7	exit Example: switch(config-router-neighbor-af)# exit	Exits BGP neighbor address family configuration mode.
Step 8	(Optional) timers <i>keepalive hold</i> 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	Exits BGP neighbor configuration mode.
Step 10	neighbor ip-address remote-as <i>as-number</i> Example: switch(config-router)# neighbor 192.168.1.2 remote-as 65535 switch(config-router-neighbor)#	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 11	inherit peer <i>template-name</i> Example:	Inherits the peer template.

	Command or Action	Purpose
	<code>switch(config-router-neighbor)# inherit peer BasePeer</code>	
Step 12	(Optional) <code>timers keepalive hold</code> Example: <code>switch(config-router-neighbor)# timers 60 120</code>	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) <code>show bgp peer-template template-name</code> Example: <code>switch(config-router-neighbor)# show bgp peer-template BasePeer</code>	Displays the peer template.
Step 14	(Optional) <code>copy running-config startup-config</code> Example: <code>switch(config-router-neighbor)# copy running-config startup-config</code>	Saves this configuration change. Use the show bgp neighbor command to see the template applied.

Example

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.

SUMMARY STEPS

1. `timers prefix-peer-timeout value`
2. `maximum-peers value`

DETAILED STEPS

	Command or Action	Purpose
Step 1	timers prefix-peer-timeout <i>value</i> Example: <pre>switch(config-router-neighbor)# timers prefix-peer-timeout 120</pre>	Configures the BGP prefix peering timeout value in router configuration mode. 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.
Step 2	maximum-peers <i>value</i> Example: <pre>switch(config-router-neighbor)# maximum-peers 120</pre>	Configures the maximum number of peers for this prefix peering in neighbor configuration mode. The range is from 1 to 1000.

Example

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)#
```

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

Configuring BGP 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 (see the [Enabling BGP](#) section).

Procedure

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal	Enters configuration mode.
Step 2	router bgp <i>autonomous-system-number</i> 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 <i>interface-name</i> remote-as { <i>as-number</i> route-map <i>map-name</i> } Example: switch(config-router)# neighbor Ethernet1/1 remote-as 65535 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 and Interface Peers for more information about using dynamic AS numbers. <i>interface-name</i> can be a range if the configuration needs to be applied to more than one interface.
Step 4	inherit peer <i>template-name</i> 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 <i>interface</i> Example: switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors e1/25 Example: switch(config-router-neighbor-af)# show bgp ipv6 unicast neighbors 3FFE:700:20:1::11	Displays information about BGP peers.
Step 7	(Optional) show ip bgp neighbors <i>interface-name</i> Example:	Displays the interface used as a BGP peer.

	Command or Action	Purpose
	switch(config-router-neighbor-af)# show ip bgp neighbors Ethernet1/1	
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.

iBGP Interface Peering Configuration for Leaf 1:

```
switch# configure terminal
switch(config)# router bgp 65000
switch(config-router)# neighbor Ethernet1/1 remote-as 65000
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.

Alternatively, beginning with Cisco NX-OS Release 10.4(2)F, you can configure BGP to authenticate route updates from peers using TCP Authentication Option (TCP AO).

Beginning with Cisco NX-OS Release 10.3(3)F, Type-6 encryption for BGP password is supported on Cisco NX-OS switches. Following encryption types are supported:

- AES based encryption
- A configurable encryption-key called as primary-key is used for encryption and decryption of secrets.

To configure BGP to use MD5 digests or TCP AO, use the following command in neighbor configuration mode:

Before you begin

- Ensure the primary-key is configured using the **key config-key ascii** *<primary_key>* command on Cisco NX-OS switches.
- For Type-6 encryption to function properly, ensure **feature password encryption aes** is enabled on Cisco NX-OS switches.
- See [Configuring TCP Authentication Option](#) to configure and use TCP keychain authentication option for BGP neighbor session authentication.

SUMMARY STEPS

1. **key config-key ascii** *<primary_key>*
2. **configure terminal**
3. **feature password encryption aes**
4. **router bgp** *AS number*
5. **template peertemplate name**
6. **password** {0 | 3 | 7 | 6} *string*
7. (Optional) **encryption re-encrypt obfuscated**
8. (Optional) **encryption delete type-6**
9. (Optional) **ao** *<Keychain-name>* [**include-tcp-options**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	key config-key ascii <i><primary_key></i> Example: <pre>switch# key config-key ascii 0123456789012345</pre>	Configures the primary-key. Note <ul style="list-style-type: none"> • Enter this command only if the primary key is not configured. • If the primary key is already configured and if you enter this command, you are actually modifying the existing primary-key value. To modify to the new value, enter the existing primary-key value when prompted.
Step 2	configure terminal Example: <pre>switch# configure terminal</pre>	Enters global configuration mode.
Step 3	feature password encryption aes Example: <pre>switch(config)# feature password encryption aes</pre>	Enables the AES password encryption.
Step 4	router bgp <i>AS number</i> Example: <pre>switch(config-router)# router bgp 1</pre>	Enters to BGP router mode.

	Command or Action	Purpose
Step 2	(Optional) <code>clear bgp {ipv4 ipv6 } {unicast multicast} ip-address soft {in out}</code> Example: <code>switch# clear bgp ip unicast 192.0.2.1 soft in</code>	Resets the BGP session without tearing down the TCP session.
Step 3	<code>clear bgp {ipv4 ipv6} {unicast multicast} ip-address soft (in out)</code> Example: <code>switch# clear bgp ip unicast 192.0.2.1 soft in</code>	Resets the BGP session without tearing down the TCP session.

Modifying the Next-Hop Address

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

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

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

SUMMARY STEPS

1. `next-hop-self`
2. `next-hop-third-party`

DETAILED STEPS

	Command or Action	Purpose
Step 1	<code>next-hop-self</code> Example: <code>switch(config-router-neighbor-af) # next-hop-self</code>	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.
Step 2	<code>next-hop-third-party</code> Example: <code>switch(config-router-neighbor-af) # next-hop-third-party</code>	Sets the next-hop address as a third-party address. Use this command for single-hop eBGP peers that do not have next-hop-self configured.

Configuring BGP Next-Hop Address Tracking

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

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

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

SUMMARY STEPS

1. `nexthop trigger-delay {critical | non-critical} milliseconds`

DETAILED STEPS

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

Configuring Next-Hop Filtering

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

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

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

SUMMARY STEPS

1. `nexthop route-map name`

DETAILED STEPS

	Command or Action	Purpose
Step 1	nexthop route-map name Example: <pre>switch(config-router-af)# nexthop route-map nextHopLimits</pre>	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.

Configuring Next-Hop Resolution via Default Route

BGP next-hop resolution allows you to specify if the IP default route is used for BGP next-hop resolution.

To configure BGP next-hop resolution, use the following command in router configuration mode:

SUMMARY STEPS

1. `[no] nexthop suppress-default-resolution`

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>[no] nexthop suppress-default-resolution</p> <p>Example:</p> <pre>switch(config-router)# nexthop suppress-default-resolution</pre>	<p>Prevents resolution of BGP next hop through the IP default route.</p> <p>When this command is enabled:</p> <ul style="list-style-type: none"> The output of the show bgp process detail command includes the following line: Use default route for nexthop resolution: No The output of the show routing clients bgp command includes the following line: Owned rnh will never resolve to 0.0.0.0/0

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
<p>next-hop-self [all]</p> <p>Example:</p> <pre>switch(config-router-af)# next-hop-self all</pre>	<p>Uses the local BGP speaker address as the next-hop address in route updates.</p> <p>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.</p>

Shrinking Next-Hop Groups When A Session Goes Down

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

This feature applies to the following BGP path failure events:

- Any single or multiple Layer 3 link failures
- Line card failures
- BFD failure detections for BGP neighbors
- Administrative shutdown of BGP neighbors (using the shutdown command)

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

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

SUMMARY STEPS

1. **neighbor-down fib-accelerate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	neighbor-down fib-accelerate Example: <pre>switch(config-router)# neighbor-down fib-accelerate</pre>	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 routes.

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:

SUMMARY STEPS

1. **dont-capability-negotiate**

DETAILED STEPS

	Command or Action	Purpose
Step 1	dont-capability-negotiate Example: <pre>switch(config-router-neighbor)# dont-capability-negotiate</pre>	Disables capabilities negotiation. You must manually reset the BGP sessions after configuring this command.

Disabling Policy Batching

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

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

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

SUMMARY STEPS

1. **disable-policy-batching**

DETAILED STEPS

	Command or Action	Purpose
Step 1	disable-policy-batching Example: <pre>switch(config-router)# disable-policy-batching</pre>	Disables the batching evaluation of prefix advertisements to all peers.

Configuring BGP Additional Paths

BGP supports sending and receiving multiple paths per prefix and advertising such paths.

Advertising the Capability of Sending and Receiving Additional Paths

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

SUMMARY STEPS

1. `[no] capability additional-paths send [disable]`
2. `[no] capability additional-paths receive [disable]`
3. `show bgp neighbor`

DETAILED STEPS

	Command or Action	Purpose
Step 1	[no] capability additional-paths send [disable] Example: <pre>switch(config-router-neighbor-af)# capability additional-paths send</pre>	Advertises the capability to send additional paths to the BGP peer. The disable option disables the advertising capability of sending additional paths. The no form of this command disables the capability of sending additional paths.
Step 2	[no] capability additional-paths receive [disable] Example: <pre>switch(config-router-neighbor-af)# capability additional-paths receive</pre>	Advertises the capability to receive additional paths from the BGP peer. The disable option disables the advertising capability of receiving additional paths. The no form of this command disables the capability of receiving additional paths.
Step 3	show bgp neighbor Example: <pre>switch(config-router-neighbor-af)# show bgp neighbor</pre>	Displays whether the local peer has advertised the additional paths send or receive capability to the remote peer.

Example

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

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

Configuring the Sending and Receiving of Additional Paths

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

SUMMARY STEPS

1. `[no] additional-paths send`
2. `[no] additional-paths receive`
3. `show bgp neighbor`

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p><code>[no] additional-paths send</code></p> <p>Example:</p> <pre>switch(config-router-af)# additional-paths send</pre>	<p>Enables the send capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled.</p> <p>The no form of this command disables the send capability.</p>
Step 2	<p><code>[no] additional-paths receive</code></p> <p>Example:</p> <pre>switch(config-router-af)# additional-paths receive</pre>	<p>Enables the receive capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled.</p> <p>The no form of this command disables the receive capability.</p>
Step 3	<p><code>show bgp neighbor</code></p> <p>Example:</p> <pre>switch(config-router-af)# show bgp neighbor</pre>	<p>Displays whether the local peer as advertised the additional paths send or receive capability to the remote peer.</p>

Example

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

```
switch# configure terminal
switch(config)# router bgp 100
```

```
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# additional-paths send
switch(config-router-af)# additional-paths receive
```

Configuring Advertised Paths

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

SUMMARY STEPS

1. `[no] set ip next-hop unchanged`
2. `[no] set path-selection { all | backup | best2 | multipaths } | advertise`
3. `show bgp {ipv4 | ipv6} unicast [ip-address | ipv6-prefix] [vrf vrf-name]`

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p><code>[no] set ip next-hop unchanged</code></p> <p>Example:</p> <pre>switch(config-route-map)# set ip next-hop unchanged</pre>	Specifies and unchanged next-hop IP address.
Step 2	<p><code>[no] set path-selection { all backup best2 multipaths } advertise</code></p> <p>Example:</p> <pre>switch(config-route-map)# set path-selection all advertise</pre>	<p>Specifies that all paths be advertised for a given prefix. You can use one of the following options:</p> <ul style="list-style-type: none"> • all—Advertises all available valid paths. • backup—Advertises paths marked as backup paths. This option requires that backup paths be enabled using the <code>additional-path install backup</code> command. • best2—Advertises the second best path, which is the best path of the remaining available paths, except the already calculated best path. • multipaths—Advertises all multipaths. This option requires that multipaths be enabled using the <code>maximum-paths</code> command. <p>Note If there are no multipaths, the <code>backup</code> and <code>best2</code> options are the same. If there are multipaths, <code>best2</code> is the first path on the list of multipaths while <code>backup</code> is the best path of all available paths, except the calculated best path and multipaths.</p> <p>The no form of this command specifies that only the best path be advertised.</p>

	Command or Action	Purpose
Step 3	show bgp {ipv4 ipv6} unicast [<i>ip-address</i> <i>ipv6-prefix</i>] [<i>vrf vrf-name</i>] Example: <pre>switch(config-route-map)# show bgp ipv4 unicast</pre>	Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

Example

This example show how to specify that all paths be advertised for the prefix list p1:

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

Configuring Additional Path Selection

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

SUMMARY STEPS

1. **[no] additional-paths selection route-map** *map-name*
2. **show bgp {ipv4 | ipv6} unicast** [*ip-address* | *ipv6-prefix*] [*vrf vrf-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	[no] additional-paths selection route-map <i>map-name</i> Example: <pre>switch(config-router-af)# additional paths selection route-map map1</pre>	Configures the capability of selecting additional paths for a prefix. The no form of this command disables the additional paths selection capability.
Step 2	show bgp {ipv4 ipv6} unicast [<i>ip-address</i> <i>ipv6-prefix</i>] [<i>vrf vrf-name</i>] Example: <pre>switch(config-route-af)# show bgp ipv4 unicast</pre>	Displays the path ID for the additional paths of a prefix and advertisement information for these paths.

Example

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

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

Configuring eBGP

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:

SUMMARY STEPS

1. **disable-connected-check**

DETAILED STEPS

	Command or Action	Purpose
Step 1	disable-connected-check Example: <pre>switch(config-router-neighbor) # disable-connected-check</pre>	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 TTL Security Hops

Perform this task to allow BGP to establish or maintain a session only if the TTL value in the IP packet header is equal to or greater than the TTL value configured for the BGP neighbor session.

Before you begin

To maximize the effectiveness of the BGP Support for TTL Security Check feature, we recommend that you configure it on each participating router. Enabling this feature secures the eBGP session in the incoming direction only and has no effect on outgoing IP packets or the remote router.



Note

- The **neighbor ebgp-multihop** command is not needed when the BGP Support for TTL Security Check feature is configured for a multihop neighbor session and should be disabled before configuring this feature.
- The effectiveness of the BGP Support for TTL Security Check feature is reduced in large-diameter multihop peerings. In the event of a CPU utilization-based attack against a BGP router that is configured for large-diameter peering, you may still need to shut down the affected neighbor sessions to handle the attack.
- This feature is not effective against attacks from a peer that has been compromised inside of the local and remote network. This restriction also includes peers that are on the network segment between the local and remote network.

SUMMARY STEPS

1. **enable**
2. **trace** *[protocol] destination*
3. **configure terminal**
4. **router bgp** *autonomous-system-number*
5. **neighbor** *ip-address*
6. **ttl-security hops** *hop-count*
7. **end**
8. **show running-config**
9. **show ip bgp neighbors** *[ip-address]*

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable Example: <pre>switch(config)# enable</pre>	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	trace <i>[protocol] destination</i> Example: <pre>switch(config)# trace ip 10.1.1.1</pre>	Discovers the routes of the specified protocol that packets will actually take when traveling to their destination. Enter the trace command to determine the number of hops to the specified peer.
Step 3	configure terminal Example: <pre>switch(config)# configure terminal</pre>	Enters global configuration mode.
Step 4	router bgp <i>autonomous-system-number</i> Example: <pre>switch(config)# router bgp 65000</pre>	Enters router configuration mode, and creates a BGP routing process.
Step 5	neighbor <i>ip-address</i> Example: <pre>switch(config)# neighbor 10.1.1.1</pre>	Configures the neighbor IP address.
Step 6	ttl-security hops <i>hop-count</i> Example: <pre>switch(config)# ttl-security hops 2</pre>	<p>Configures the maximum number of hops that separate two peers.</p> <p>The hop-count argument is set to the number of hops that separate the local and remote peer. If the expected TTL value in the IP packet header is 254, then the number 1 should be configured for the hop-count argument. The range of values is a number from 1 to 254.</p> <p>When the BGP Support for TTL Security Check feature is enabled, BGP will accept incoming IP packets with a TTL value that is equal to or greater than the expected TTL value. Packets that are not accepted are discarded.</p>

	Command or Action	Purpose
		The example configuration sets the expected incoming TTL value to at least 253, which is 255 minus the TTL value of 2, and this is the minimum TTL value expected from the BGP peer. The local router will accept the peering session from the 10.1.1.1 neighbor only if it is one or two hops away.
Step 7	end Example: <pre>switch(config)# end</pre>	Exits router configuration mode and enters privileged EXEC mode.
Step 8	show running-config Example: <pre>switch(config)# show running-config begin bgp</pre>	(Optional) Displays the contents of the currently running configuration file. The output of this command displays the configuration of the neighbor ttl-security command for each peer under the BGP configuration section of output. That section includes the neighbor address and the configured hop count. Note Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS IP Routing: BGP Command Reference.
Step 9	show ip bgp neighbors [ip-address] Example: <pre>switch(config)# show ip bgp neighbors 10.4.9.5</pre>	(Optional) Displays information about the TCP and BGP connections to neighbors. This command displays "External BGP neighbor may be up to number hops away" when the BGP Support for TTL Security Check feature is enabled. The number value represents the hop count. It is a number from 1 to 254. Note Only the syntax applicable to this task is used in this example. For more details, see the Cisco IOS IP Routing: BGP Command Reference.

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.



Note This configuration is not supported for BGP interface peering.

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

SUMMARY STEPS

1. `ebgp-multihop ttl-value`

DETAILED STEPS

	Command or Action	Purpose
Step 1	ebgp-multihop <i>tvl-value</i> Example: <pre>switch(config-router-neighbor) # ebgp-multihop 5</pre>	Configures the eBGP TTL value for eBGP multihop. The range is from 2 to 255. You must manually reset the BGP sessions after using this command.

Disabling a Fast External Fallover

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

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

SUMMARY STEPS

1. **no fast-external-fallover**

DETAILED STEPS

	Command or Action	Purpose
Step 1	no fast-external-fallover Example: <pre>switch(config-router) # no fast-external-fallover</pre>	Disables a fast external fallover for eBGP peers. This command is enabled by default.

Limiting the AS-path Attribute

You can configure eBGP to discard routes that have 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:

SUMMARY STEPS

1. **maxas-limit *number***

DETAILED STEPS

	Command or Action	Purpose
Step 1	maxas-limit <i>number</i> Example: <pre>switch(config-router) # maxas-limit 50</pre>	Discards eBGP routes that have a number of AS-path segments that exceed the specified limit. The range is from 1 to 2000.

Configuring Local AS Support

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

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

Furthermore, the remote peer's ASN configured with the remote-as command cannot be identical to the local device's ASN configured with the local-as command.

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

SUMMARY STEPS

1. `local-as number [no-prepend [replace-as [dual-as]]]`

DETAILED STEPS

	Command or Action	Purpose
Step 1	local-as number [no-prepend [replace-as [dual-as]]] Example: <pre>switch(config-router-neighbor)# local-as 1.1</pre>	Configures eBGP to prepend the local AS <i>number</i> to the AS_PATH attribute. The AS <i>number</i> 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.

Example

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

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

Configuring AS Confederations

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

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

SUMMARY STEPS

1. `confederation identifier as-number`
2. `bgp confederation peers as-number [as-number2...]`

DETAILED STEPS

	Command or Action	Purpose
Step 1	confederation identifier <i>as-number</i> Example: <pre>switch(config-router)# confederation identifier 4000</pre>	In router configuration mode, this command configures a BGP confederation identifier. The command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 2	bgp confederation peers <i>as-number [as-number2...]</i> Example: <pre>switch(config-router)# bgp confederation peers 5 33 44</pre>	In router configuration mode, this command configures the autonomous systems that belong to the AS confederation. The command specifies a list of autonomous systems that belong to the confederation and it triggers an automatic notification and session reset for the BGP neighbor sessions.

Configuring Route Reflector

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

Before you begin

You must enable BGP.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example:	Enters global configuration mode.

	Command or Action	Purpose
	<code>switch# configure terminal</code>	
Step 2	router bgp <i>as-number</i> Example: <code>switch(config)# router bgp 65535</code> <code>switch(config-router)#</code>	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	cluster-id <i>cluster-id</i> Example: <code>switch(config-router)# cluster-id 192.0.2.1</code>	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: <code>switch(config-router)# address-family ipv4 unicast</code> <code>switch(config-router-af)#</code>	Enters router address family configuration mode for the specified address family.
Step 5	(Optional) client-to-client reflection Example: <code>switch(config-router-af)# client-to-client reflection</code>	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: <code>switch(config-router-af)# exit</code> <code>switch(config-router)#</code>	Exits router address configuration mode.
Step 7	neighbor <i>ip-address remote-as as-number</i> Example: <code>switch(config-router)# neighbor 192.0.2.10 remote-as 65535</code> <code>switch(config-router-neighbor)#</code>	Configures the IP address and AS number for a remote BGP peer.
Step 8	address-family { ipv4 ipv6 } { unicast multicast } Example: <code>switch(config-router-neighbor)# address-family ipv4 unicast</code> <code>switch(config-router-neighbor-af)#</code>	Enters neighbor address family configuration mode for the unicast IPv4 address family.
Step 9	route-reflector-client Example: <code>switch(config-router-neighbor-af)# route-reflector-client</code>	Configures the device as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 10	(Optional) show bgp { ipv4 ipv6 } { unicast multicast } neighbors Example:	Displays the BGP peers.

	Command or Action	Purpose
	<pre>switch(config-router-neighbor-af) # show bgp ipv4 unicast neighbors</pre>	
Step 11	<p>(Optional) copy running-config startup-config</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # copy running-config startup-config</pre>	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 Next-Hops on Reflected Routes Using an Outbound Route-Map

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



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

Before you begin

You must enable BGP (see the [Enabling BGP](#) section).

Ensure that you are in the correct VDC (or use the **switchto vdc** command).

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

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

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

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	router bgp <i>as-number</i> Example: switch(config)# router bgp 200 switch(config-router)#	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	neighbor <i>ip-address</i> remote-as <i>as-number</i> Example: switch(config-router)# neighbor 192.0.2.12 remote-as 200 switch(config-router-neighbor)#	Configures the IP address and AS number for a remote BGP peer.
Step 4	(Optional) update-source <i>interface number</i> Example: switch(config-router-neighbor)# update-source loopback 300	Specifies and updates the source of the BGP session.
Step 5	address-family { ipv4 ipv6 } { unicast multicast } Example: switch(config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	Enters router address family configuration mode for the specified address family.

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

Example

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

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

Configuring Route Dampening

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

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

SUMMARY STEPS

1. **dampening** [*{half-life reuse-limit suppress-limit max-suppress-time | route-map map-name}*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	dampening [<i>{half-life reuse-limit suppress-limit max-suppress-time route-map map-name}</i>] Example: <pre>switch(config-router-af) # dampening route-map bgpDamp</pre>	Disables capabilities negotiation. The parameter values are as follows: <ul style="list-style-type: none"> • <i>half-life</i>—The range is from 1 to 45. • <i>reuse-limit</i>—The range is from 1 to 20000. • <i>suppress-limit</i>—The range is from 1 to 20000. • <i>max-suppress-time</i>—The range is from 1 to 255.

Configuring Load Sharing and ECMP

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

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

SUMMARY STEPS

1. **maximum-paths** [**ibgp**] *maxpaths*

DETAILED STEPS

	Command or Action	Purpose
Step 1	maximum-paths [ibgp] <i>maxpaths</i> Example: <pre>switch(config-router-af) # maximum-paths 8</pre>	Configures the maximum number of equal-cost paths for load sharing. The default is 1.

Unequal Cost Multipath (UCMP) over BGP

UCMP is also known as Weighted ECMP. It is a mechanism that allows multiple routes to the same destination with different weights per next-hop and load-balances the routed traffic over those multiple next-hops. The basic UCMP works for most of the customers' requirements. The load entropy is the best way to maximize the link usage efficiency.

Often, the application distribution in the network can be unbalanced. The new clusters roll in at different over-subscription rates than the old clusters. The new clusters have powerful servers than the old clusters and they are capable of handling more load per CPU. As the network is not perfect, some control over routing behavior is needed. You can configure Weighted ECMP over BGP for balancing the traffic load and for administering control over the routing behavior.



Note The Link-Bandwidth Extended Community must be advertised across eBGP sessions, although it is defined as a non-transitive attribute.

Next-hop-self must strip the Link-Bandwidth Extended Community from advertisements.

Enabling UCMP over BGP

The solution for the unequal distribution of the resources and sub-optimal traffic distribution use-cases is to configure Weighted ECMP over BGP. You can inject the routes (from the host or the controller) and signal a weight for each instance. You can then aggregate the weights across the infrastructure and deliver the traffic in the direct proportion to the application deployment distribution.

Guidelines and Limitations for UCMP over BGP

- BGP uses the Link-Bandwidth Extended Community defined in the draft-ietf-idr-link-bandwidth-06.txt to implement the weighted ECMP feature. The Link-Bandwidth Extended Community is advertised across eBGP sessions, although it's defined as a non-transitive attribute, as long as next-hop is unchanged.
- You can accept Link-Bandwidth Extended Community from both iBGP and eBGP peers.
- For weights programming, the Link-Bandwidth Extended Community has the link bandwidth encoded in bytes/second, as a four byte floating point integer, that is normalized between 0 and 1000 before downloading to RIB.
- The hardware ECMP width is fixed as 64 in size.

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:

SUMMARY STEPS

1. **maximum-prefix** *maximum* [*threshold*] [**restart time** | **warning-only**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	maximum-prefix <i>maximum</i> [<i>threshold</i>] [restart time warning-only] Example: <pre>switch(config-router-neighbor-af)# maximum-prefix 12</pre>	Configures the maximum number of prefixes from a peer. The parameter ranges are as follows: <ul style="list-style-type: none"> • <i>maximum</i>—The range is from 1 to 300000. • <i>threshold</i>—The range is from 1 to 100 percent. The default is 75 percent. • <i>time</i>—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 is exceeded.

Configuring DSCP

You can configure a differentiated services code point (DSCP) for a neighbor. You can specify a DSCP value for locally originated packets for IPv4 or IPv6.

To configure the DSCP value, use the following command in neighbor configuration mode:

SUMMARY STEPS

1. **dscp** *dscp_value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	dscp <i>dscp_value</i> Example: <pre>switch(config-router-neighbor)# dscp 63</pre> Below is an example of the corresponding show command: <pre>show ipv6 bgp neighbors BGP neighbor is 10.1.1.1, remote AS 0, unknown link, Peer index 4 BGP version 4, remote router ID 0.0.0.0 BGP state = Idle, down for 00:13:34, retry in 0.000000 DSCP (DiffServ CodePoint): 0 Last read never, hold time = 180, keepalive interval is 60 seconds</pre>	Sets the differentiated services code point (DSCP) value for the neighbor. The DSCP value can be a number from 0 to 63, or it can be one of the following keywords: ef , af11 , af12 , af13 , af21 , af22 , af23 , af31 , af32 , af33 , af41 , af42 , af43 , cs1 , cs2 , cs3 , cs4 , cs5 , cs6 , or cs7 . The default value is cs6.

Configuring Dynamic Capability

You can configure dynamic capability for a BGP peer.

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

SUMMARY STEPS

1. **dynamic-capability**

DETAILED STEPS

	Command or Action	Purpose
Step 1	dynamic-capability Example: <pre>switch(config-router-neighbor)# dynamic-capability</pre>	Enables dynamic capability. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

Configuring Aggregate Addresses

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

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

SUMMARY STEPS

1. **aggregate-address** *ip-prefix/length* [**as-set**] [**summary-only**] [**advertise-map** *map-name*] [**attribute-map** *map-name*] [**suppress-map** *map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	aggregate-address <i>ip-prefix/length</i> [as-set] [summary-only] [advertise-map <i>map-name</i>] [attribute-map <i>map-name</i>] [suppress-map <i>map-name</i>] Example: <pre>switch(config-router-af)# aggregate-address 192.0.2.0/8 as-set</pre>	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: <ul style="list-style-type: none"> • 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.

	Command or Action	Purpose
		<ul style="list-style-type: none"> • The attribute-map keyword and argument specify the route map used to select attribute information from the aggregate. • The suppress-map keyword and argument conditionally filter more specific routes. If you specify the suppress-map option while performing a BGP route aggregation, you can set the community attribute for a BGP route update. This option enables you to set community attributes on the more-specific routes. • The suppress-map keyword and argument conditionally filter more specific routes. If you specify the suppress-map option while performing a BGP route aggregation, you can either suppress certain more-specific routes from being advertised to its peers, or decide to advertise the more-specific routes with some community attributes set on them, depending upon the suppress-map route-map configuration. A route-map configured with only match clauses will suppress the more-specific routes that satisfy the match criteria. However, if a route-map is configured with match and set clauses, then the routes satisfying the match criteria will be advertised with the appropriate attributes as modified by the route-map. The second option enables you to set community attributes on the more-specific routes.

Suppressing BGP Routes

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

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

SUMMARY STEPS

1. **suppress-fib-pending**

DETAILED STEPS

	Command or Action	Purpose
Step 1	suppress-fib-pending Example: <pre>switch(config-router)# suppress-fib-pending</pre>	Suppresses newly learned BGP routes (IPv4 or IPv6) from being advertised to downstream BGP neighbors until the routes have been programmed in the hardware.

Configuring BGP Conditional Advertisement

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

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

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

Before you begin

You must enable BGP(see the [Enabling BGP](#) section).

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters configuration mode.
Step 2	router bgp <i>as-number</i> Example: <pre>switch(config)# router bgp 65535 switch(config-router)#</pre>	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	neighbor <i>ip-address</i> remote-as <i>as-number</i> Example: <pre>switch(config-router)# neighbor 192.168.1.2 remote-as 65534 switch(config-router-neighbor)#</pre>	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

	Command or Action	Purpose
Step 4	address-family {ipv4 ipv6} {unicast multicast} Example: <pre>switch(config-router-neighbor) # address-family ipv4 multicast switch(config-router-neighbor-af) #</pre>	Enters address family configuration mode.
Step 5	advertise-map adv-map {exist-map exist-rmap non-exist-map nonexist-rmap} Example: <pre>switch(config-router-neighbor-af) # advertise-map advertise exist-map exist</pre>	<p>Configures BGP to conditionally advertise routes based on the two configured route maps:</p> <ul style="list-style-type: none"> • <i>adv-map</i>—Specifies a route map with match statements that the route must pass before BGP passes the route to the next route map. The <i>adv-map</i> is a case-sensitive, alphanumeric string up to 63 characters. • <i>exist-rmap</i>—Specifies a route map with match statements for a prefix list. A prefix in the BGP table must match a prefix in the prefix list before BGP advertises the route. The <i>exist-rmap</i> is a case-sensitive, alphanumeric string up to 63 characters. • <i>nonexist-rmap</i>—Specifies a route map with match statements for a prefix list. A prefix in the BGP table must not match a prefix in the prefix list before BGP advertises the route. The <i>nonexist-rmap</i> is a case-sensitive, alphanumeric string up to 63 characters. <p>Note For BGP conditional advertisement feature, ensure that the "le" or "ge" statements are not used on prefix-list when associated to exist or nonexist map.</p>
Step 6	(Optional) show bgp {ipv4 ipv6} {unicast multicast} neighbors Example: <pre>switch(config-router-neighbor-af) # show ip bgp neighbor</pre>	Displays information about BGP and the configured conditional advertisement route maps.
Step 7	(Optional) copy running-config startup-config Example: <pre>switch(config-router-neighbor-af) # copy running-config startup-config</pre>	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

You must enable BGP.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **address-family** {*ipv4* | *ipv6*} {*unicast* | *multicast*}
4. **address-family** {*ipv4* | *ipv6*} {*unicast* | *multicast*}
5. **redistribute** {*direct* | {*eigrp* | *isis* | *ospf* | *ospfv3* | *rip*} *instance-tag* | *static* | *icmpv6*} **route-map** *map-name*
6. (Optional) **default-metric** *value*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	router bgp <i>as-number</i> Example: <pre>switch(config)# router bgp 65535 switch(config-router)#</pre>	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	address-family { <i>ipv4</i> <i>ipv6</i> } { <i>unicast</i> <i>multicast</i> }	Enters address family configuration mode.
	Example: <pre>switch(config-router)# address-family vpnv4 unicast switch(config-router-af)#</pre>	

	Command or Action	Purpose
Step 4	address-family { ipv4 ipv6 } { unicast multicast } Example: switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Enters address-family configuration mode.
Step 5	redistribute { direct { eigrp isis ospf ospfv3 rip } <i>instance-tag</i> static icmpv6 } route-map <i>map-name</i> Example: switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap	Redistributes routes from other protocols into BGP. Beginning with Cisco NX-OS Release 10.3(3)F, the keyword icmpv6 is supported to redistribute icmpv6 routes from other protocols into BGP.
Step 6	(Optional) default-metric <i>value</i> Example: switch(config-router-af)# default-metric 33	Generates a default route into BGP.
Step 7	(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
```

DMZ Link Bandwidth

The DMZ Link Bandwidth feature is used to enable traffic load balancing towards a BGP learnt route reachable via multiple autonomous system exit links. The load balancing is done proportional to the bandwidth of these links.

Link Bandwidth Extended Community is used to carry the bandwidth of the link between two directly connected (single hop) eBGP peers. A nexus device will attach this extended community to BGP routes received from a directly connected eBGP neighbor if the `dmz-link-bandwidth` command is configured under that neighbor's address-family mode. This extended community is then propagated to iBGP peers when extended community exchange is enabled with the `send-community extended` or `send-community both` command. This attribute is used as a load sharing value relative to other paths in forwarding.

In addition, user may want to forcefully change the Link Bandwidth Extended Community for routes received from a BGP peer. They may also only want to set this extended community for only a subset of routes received

from the peer. They can achieve that by configuring an inbound route-map towards the peer and configure 'set extcommunity bandwidth <1-4000000>' under it.

Guidelines and Limitations

BGP DMZ Link Bandwidth

Consider the following guidelines and limitations before configuring the Link Bandwidth feature:

- The **dmz-link-bandwidth** command can be configured only under IPv4 unicast and IPv6 unicast address families under a BGP neighbor.
- It will only attach the Link Bandwidth Extended Community to routes received from directly connected BGP neighbors. It will not do so for BGP multi-hop neighbors.
- It can be configured under both global mode and VRF mode.
- BGP multipath load balancing must be configured under the address-family using **maximum-paths** command for this feature to be enabled.
- BGP extended community exchange must be enabled between iBGP neighbors to which the Link Bandwidth Extended Community is to be advertised.
- Link Bandwidth Extended Community will be seamlessly carried to routes leaked from one VRF to another VRF.

Configuring BGP DMZ Link Bandwidth

Beginning with Cisco NX-OS Release 10.5(1)F, you can configure this feature.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *as-number*
3. **address-family** [**ipv4|ipv6**] **unicast**
4. **maximum-paths** *max-path*
5. **template peer** *peer-template-name*
6. **address-family** [**ipv4 | ipv6**] **unicast**
7. **dmz-link-bandwidth**
8. **neighbor** *neighbor*
9. **remote-as** *remote-as*
10. **address-family** [**ipv4 | ipv6**] **unicast**
11. **dmz-link-bandwidth**
12. **route-map** *name* **permit** *route*
13. **set extcommunity bandwidth** <1-4000000>
14. **neighbor** *neighbor* **address-family** [**ipv4 | ipv6**] **unicast** **route-map** *name* **in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal	Enters global configuration mode.
Step 2	router bgp <i>as-number</i> Example: switch# configure terminal	Enters router configuration mode to create or configure a BGP routing process.
Step 3	address-family [ipv4 ipv6] unicast Example: switch(config)# address-family ipv4 unicast	Configures address family IPv4 or IPv6 unicast.
Step 4	maximum-paths <i>max-path</i> Example: switch(config)# maximum-paths 10	Enable BGP multi-path under address-family.
Step 5	template peer <i>peer-template-name</i> Example: switch(config)# template peer host_peer	Enters template mode and configures peer parameter.
Step 6	address-family [ipv4 ipv6] unicast Example: switch(config)# address-family ipv4 unicast	Configures the address family for IPv4 or IPv6.
Step 7	dmz-link-bandwidth Example: switch(config)# dmz-link-bandwidth	Configures BGP to consider load balancing some traffic towards this directly connected peer by attaching link bandwidth extended community to routes received from it.
Step 8	neighbor <i>neighbor</i> Example: switch(config)# neighbor 1.1.1.1 or switch(config)# neighbor 11::1	Configure BGP neighbor.
Step 9	remote-as <i>remote-as</i> Example: switch(config)# remote-as 100	Specify Autonomous System Number of the neighbor .
Step 10	address-family [ipv4 ipv6] unicast Example: switch(config)# address-family ipv4 unicast	Configures the address family IPv4 or IPv6 unicast.

	Command or Action	Purpose
Step 11	dmz-link-bandwidth Example: <pre>switch(config)# dmz-link-bandwidth</pre>	Configures BGP to consider load balancing some traffic towards this directly connected peer by attaching link bandwidth extended community to routes received from it.
Step 12	route-map name permit route Example: <pre>switch(config)# route-map change_link_bandwidth permit 10</pre>	Configure route-map.
Step 13	set extcommunity bandwidth <1-400000> Example: <pre>switch(config-route-map)# set extcommunity bandwidth 1000</pre>	Configure route-map to set link bandwidth extended community.
Step 14	neighbor neighbor address-family [ipv4 ipv6] unicast route-map name in Example: <pre>switch(config-router)# neighbor 1.1.1.1 switch (config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)# route-map change_link_bandwidth in</pre>	Configure neighbor to be attached to an inbound route-map.

Configuration Examples for BGP DMZ Link Bandwidth

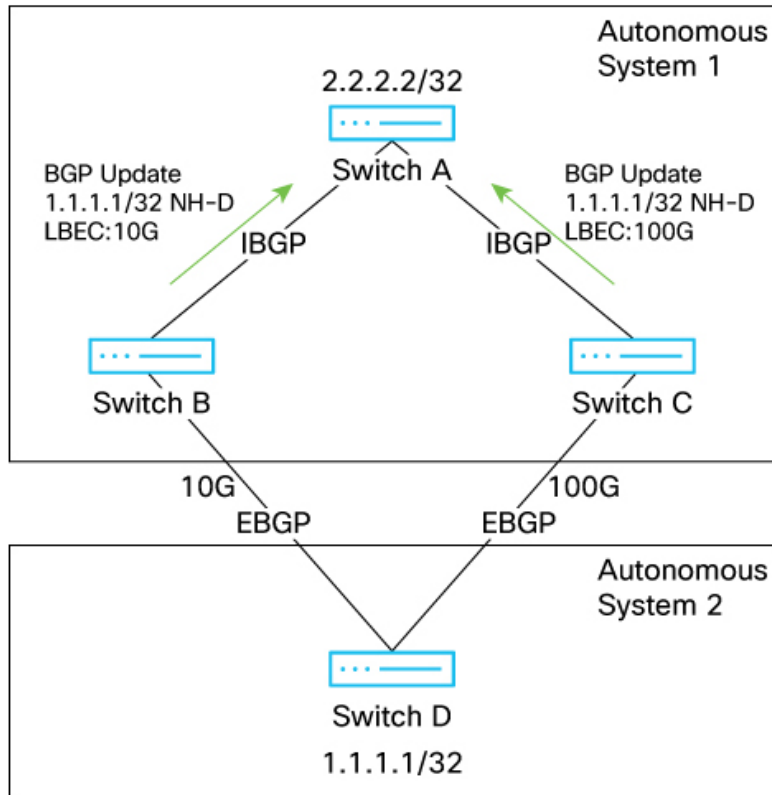
In the following example, AS 1 is attached to AS 2 through two unequal bandwidth links. B-D link is 10G and C-D link is 100G. B-D link is 10G and C-D link is 100G. A-B and A-C connected by iBGP session. B-D and C-D connected by eBGP Session.

If you want traffic to be proportionally load balanced according to these link bandwidths that is A should send 10 times the outgoing traffic to C as compared to B, configure `dmz-link-bandwidth` command on B and C towards eBGP neighbor D. B will package B-D bandwidth into Link Bandwidth Extended Community (LBEC) and attach it to the BGP path for route entry 1.1.1.1/32. Similarly, C will package C-D bandwidth into LBEC and attach it to the BGP path for route entry 1.1.1.1/32.

B and C will advertise the route to iBGP peer A along with LBEC.

On A, BGP will program forwarding with hash 10/110 for NH-B and 100/110 for NH-C.

Figure 5: DMZ Link Bandwidth Configuration



524219

Switch B to D and B to A Configuration

```
router bgp 1
neighbor D
  address-family ipv4|v6 unicast
  dmz-link-bandwidth
neighbor A
  address-family ipv4|v6 unicast
  send-community extended
```

Switch C to D and C to A Configuration

```
router bgp 1
neighbor D
  address-family ipv4|v6 unicast
  dmz-link-bandwidth
neighbor A
  address-family ipv4|v6 unicast
  send-community extended
```

Switch A Control Plane and Data Plane State

BGP table state:

```
1.1.1.1/32
```

```
NH-B    LBEC: 10G
NH-C    LBEC: 100G
```

Forwarding state:

```
1.1.1.1/32
NH-B    hash 10:110
NH-C    hash 100:110
```

Configuring Unequal Cost Multipath (UCMP) Using Link Bandwidth Extended Community

Before you begin

See [Configuring BGP DMZ Link Bandwidth](#). That feature must be first configured at edge devices for this feature to work. This means that at the edge devices, the Link Bandwidth Extended Community must be attached to a BGP route received from a directly connected ebgp peer by either configuring **dmz-link-bandwidth** command or by configuring an inbound route-map with **set extcommunity link-bandwidth <1-4000000>** command. Until that happens, none of the functionality described in this section will work.

Beginning with Cisco NX-OS Release 10.5(1)F, a BGP speaker has the ability to convey to its directly connected BGP neighbor the cumulative bandwidth available from itself to a BGP learnt route if the command 'link-bandwidth cumulative' is configured under the neighbor's address-family mode. It does that by leveraging the Link Bandwidth Extended Community. It will advertise the BGP route with the value n inserted in this extended community, where $n = \min(\text{Sum of bandwidth obtained from Link Bandwidth Extended Community of all available multi-paths, Bandwidth of link towards the neighbor to which advertisement is being sent})$.

Guidelines and Limitations

Consider the following guidelines and limitations before configuring the BGP UCMP feature:

- The **link-bandwidth cumulative** command can be configured only under IPv4 unicast and IPv6 unicast address families under a BGP neighbor.
- It will only take effect to towards directly connected BGP neighbors.
- This command will only take effect if all of the available multi-paths have the Link Bandwidth Extended Community.
- It can be configured under both global mode and vrf mode.
- Link Bandwidth Extended Community will be seamlessly carried to routes leaked from one VRF to another VRF.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp *as-number***
3. **neighbor *neighbor***
4. **remote-as *remote-as***
5. **address-family [*ipv4* | *ipv6*] unicast**
6. **send-community extended**

7. link-bandwidth cumulative

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal	Enters global configuration mode.
Step 2	router bgp <i>as-number</i> Example: switch(config)# router bgp 120	Enters router configuration mode to create or configure a BGP routing process.
Step 3	neighbor <i>neighbor</i> Example: switch(config)# neighbor 1.1.1.1 or switch(config)# neighbor 11::1	Configure BGP neighbor.
Step 4	remote-as <i>remote-as</i> Example: switch(config)# remote-as 100	Specify Autonomous System Number of the neighbor.
Step 5	address-family [ipv4 ipv6] unicast Example: switch(config)# address-family ipv4 unicast	Configures the address family IPv4 or IPv6 unicast.
Step 6	send-community extended Example: switch(config)# send-community extended	Configures sending of BGP extended community towards this neighbor.
Step 7	link-bandwidth cumulative Example: switch(config)# link-bandwidth cumulative	Sends cumulative link bandwidth towards the neighbor. This configuration doesn't advertise the cumulative link bandwidth to the neighbor if any of the multi-paths doesn't have the link bandwidth extended community.

Configuration Example

Configuration Example

In the following figures:

- Single hop directly connected ebgp devices in 4 clos layers.
- All links are 100G, except B-D and B-E, which are 10G. This means traffic from B to destination 1.1.1.1 is bottle necked at a 20G link bandwidth.

- User wants end-to-end traffic load balancing should account for this lower link bandwidth.
- On each device in Layer T1, configure command **dmz-link-bandwidth** towards every BGP neighbor in Layer T0.
- On each device in Layer T1, configure command **link-bandwidth cumulative** towards every BGP peer in layer T2.
- On each device in Layer T2, configure command **link-bandwidth cumulative** towards every BGP peer in layer T1 and T3.
- On each device in Layer T3, configure command **link-bandwidth cumulative** towards every BGP peer in layer T2.
- Command **dmz-link-bandwidth** will cause switch D to package the bandwidth of link D-F into Link Bandwidth Extended Community (LBEC) and attach it to the corresponding BGP path for route entry 1.1.1.1/32.
- Command **dmz-link-bandwidth** will cause switch E package the bandwidth of link E-F into LBEC and attach it to the corresponding BGP path for route entry 1.1.1.1/32.
- Command **link-bandwidth cumulative** will cause switches A, B, C, D, and E to insert bandwidth **n** into LBEC while advertising BGP update to the peers under which it is configured.
- The value **n** is calculated by using the formula $\min(\text{Sum of LBEC of all multi-paths, bandwidth towards peer where update is to be advertised.})$
- Propagation of LBEC in BGP update will cause forwarding on all devices to be programmed with proportional hashes
- LBEC will be dynamically recalculated in case of link failures.

Figure 6: Configuration Example 1

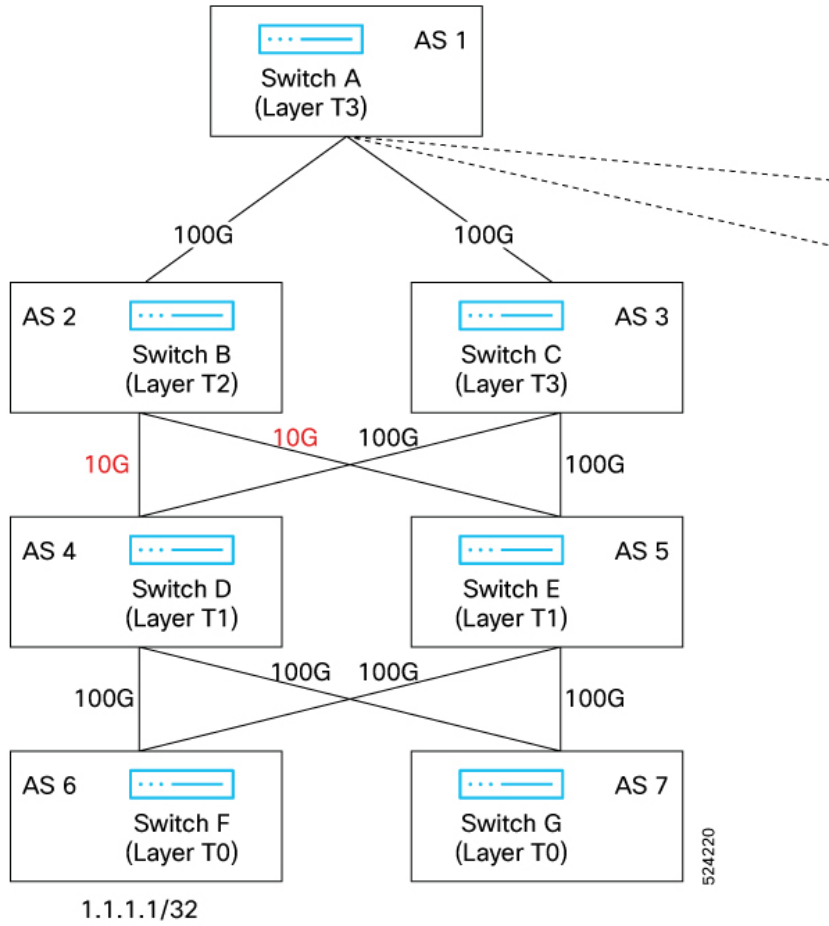
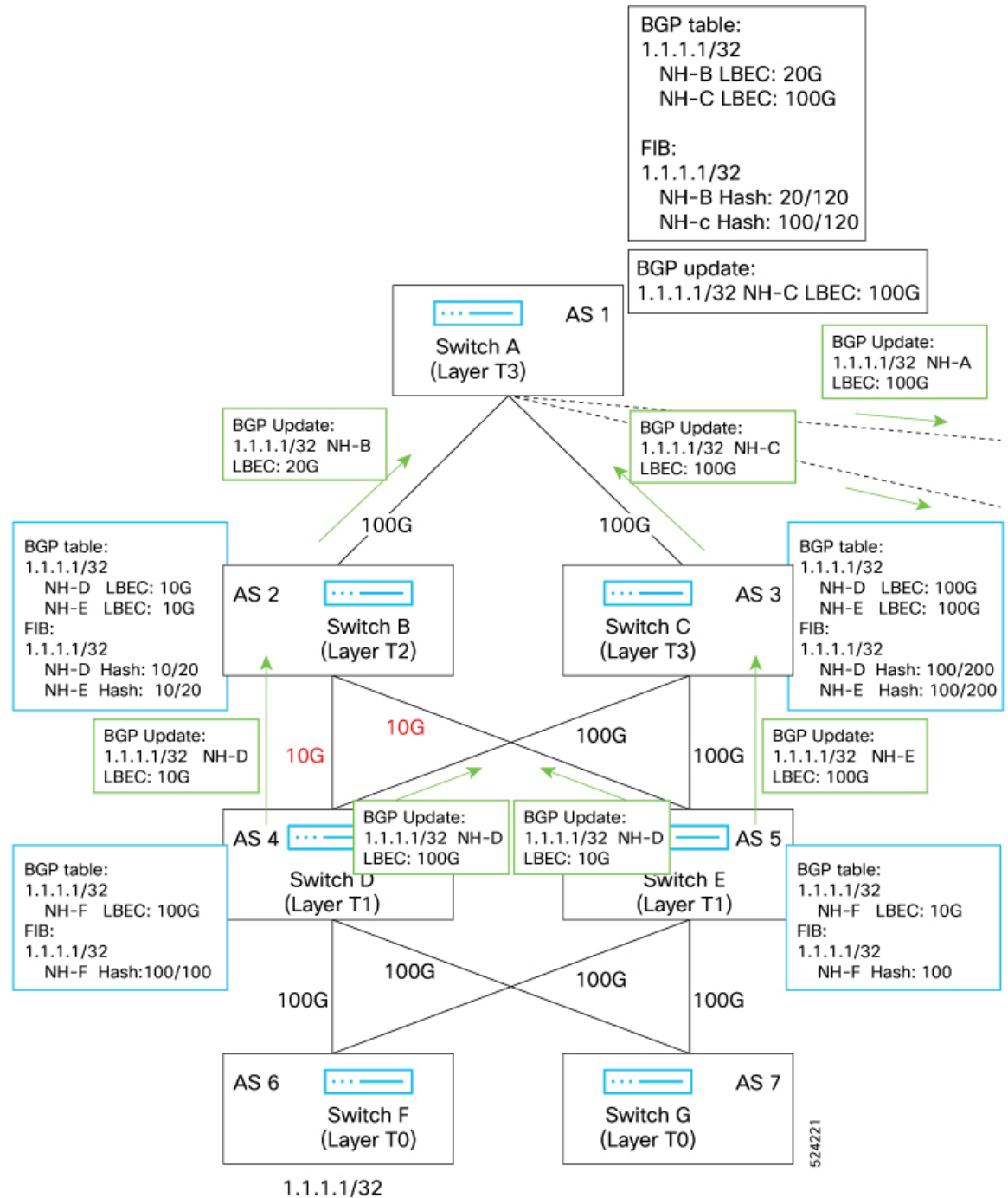


Figure 7: Configuration Example 2



Configuring A to B and C

```
router bgp 1
neighbor B
address-family ipv4|v6 unicast
```



```

        link-bandwidth cumulative
        send-community extended
neighbor C
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor B'
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor C'
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended

```

Configuring B to D, A, and E

```

router bgp 1
neighbor B
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor C
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor B'
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor C'
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended

```

Configuring C to A, D, and E

```

router bgp 3
neighbor A
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor D
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor E
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended

```

Configuring D to F, B, and C

```

router bgp 4
neighbor F
    address-family ipv4|v6 unicast
        dmz-link-bandwidth
neighbor B
    address-family ipv4|v6 unicast
        link-bandwidth cumulative
        send-community extended
neighbor C

```

```
address-family ipv4|v6 unicast
  link-bandwidth cumulative
  send-community extended
```

Configuring E to F, B, and C

```
router bgp 5
  neighbor F
    address-family ipv4|v6 unicast
      dmz-link-bandwidth
  neighbor B
    address-family ipv4|v6 unicast
      link-bandwidth cumulative
      send-community extended
  neighbor C
    address-family ipv4|v6 unicast
      link-bandwidth cumulative
      send-community extended
```

Verifying Configuration

Use the following commands to verify configuration:

- Use the following command to see if the dmz-link-bandwidth command is enabled towards a peer

```
show bgp ipv4 unicast neighbors 192.168.11.2 | i i link
dmz-link-bandwidth is enabled
```

- Use the following command to see if the link-bandwidth cumulative command is enabled towards a peer

```
show bgp ipv4 unicast neighbors 10.1.1.2 | i i link
link-bandwidth cumulative is enabled
```

- Use the following command to confirm if BGP path has Link Bandwidth Extended Community

```
show bgp ipv4 unicast 1.1.1.1/32
BGP routing table information for VRF default, address family IPv4 Unicast
BGP routing table entry for 1.1.1.1/32, version 403 Paths: (1 available, best #1) Flags:
  (0x8000001a) (high32 0x002000) on xmit-list, is in urib, is best urib route, is in HW

Advertised path-id 1 Path type: external, path is valid, is best path, no labeled
nexthop, in rib

AS-Path: 10 33299 51178 47751 {27016} , path sourced external to AS 192.168.11.2 (metric
  0) from 192.168.11.2 (192.168.11.2) Origin EGP, MED 2219, localpref 100, weight 0
Community: 1:1 Extcommunity: LB:1:125000000

Path type: external, path is valid, is multi-path, no labeled nexthop, in rib
AS-Path: 10 33299 51178 47751 {27016} , path sourced external to AS 192.168.11.3 (metric
  0) from 192.168.11.2 (192.168.11.2) Origin EGP, MED 2219, localpref 100, weight 0
Community: 1:1 Extcommunity: LB:1:250000000
```

- Use the following command to confirm if routing table has the hashing ratios

```
show ip route 1.1.1.1/32 detail

100.1.1.1/32, ubest/mbest: 1/0 *via 192.168.11.2, [20/2219], 00:14:22, bgp-1, bw:333,
external, tag 10 client-specific data: 10 recursive next hop: 192.168.11.2/32 extended
route information: BGP origin AS 0 BGP peer AS 10
100.1.1.2/32, ubest/mbest: 1/0 *via 192.168.11.3, [20/2219], 00:14:22, bgp-1, bw:666,
external, tag 10 client-specific data: 10 recursive next hop: 192.168.11.3/32 extended
route information: BGP origin AS 0 BGP peer AS 10
```

Advertising the Default Route

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

Before you begin

You must enable BGP (see the [Enabling BGP](#) section).

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	route-map allow permit Example: <pre>switch(config)# route-map allow permit switch(config-route-map)#</pre>	Enters router map configuration mode and defines the conditions for redistributing routes.
Step 3	exit Example: <pre>switch(config-route-map)# exit switch(config)#</pre>	Exits router map configuration mode.
Step 4	ip route <i>ip-address network-mask null null-interface-number</i> Example: <pre>switch(config)# ip route 192.0.2.1 255.255.255.0 null 0</pre>	Configures the IP address.
Step 5	router bgp <i>as-number</i> Example:	Enters BGP mode and assigns the AS number to the local BGP speaker.

	Command or Action	Purpose
	<pre>switch(config)# router bgp 65535 switch(config-router)#</pre>	
Step 6	address-family {ipv4 ipv6} unicast Example: <pre>switch(config-router)# address-family ipv4 unicast switch(config-router-af)#</pre>	Enters address-family configuration mode.
Step 7	default-information originate Example: <pre>switch(config-router-af)# default-information originate</pre>	Advertises the default route.
Step 8	redistribute static route-map allow Example: <pre>switch(config-router-af)# redistribute static route-map allow</pre>	Redistributes the default route.
Step 9	(Optional) copy running-config startup-config Example: <pre>switch(config-router-af)# copy running-config startup-config</pre>	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	<p>[no] path-attribute treat-as-withdraw [<i>value</i> range start end] in</p> <p>Example:</p> <pre>switch#(config-router)# neighbor 10.20.30.40 switch(config-router-neighbor)# path-attribute treat-as-withdraw 100 in</pre> <p>Example:</p> <pre>switch#(config-router)# neighbor 10.20.30.40 switch(config-router-neighbor)# path-attribute treat-as-withdraw range 21 255 in</pre>	<p>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.</p> <p>This command is also supported for BGP template peers and BGP template peer sessions.</p>

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	<p>[no] path-attribute discard [<i>value</i> range start end] in</p> <p>Example:</p> <pre>switch#(config-router)# neighbor 10.20.30.40 switch(config-router-neighbor)# path-attribute discard 100 in</pre> <p>Example:</p> <pre>switch#(config-router)# neighbor 10.20.30.40 switch(config-router-neighbor)# path-attribute discard range 100 255 in</pre>	<p>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.</p> <p>This command is also supported for BGP template peers and BGP template peer sessions.</p> <p>Note When the same path attribute is configured for both discard and treat-as-withdraw, treat-as-withdraw has a higher priority.</p>

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 reen able enhanced error handling, use the following command in router configuration mode:

Procedure

	Command or Action	Purpose
Step 1	<p>[no] enhanced-error</p> <p>Example:</p> <pre>switch(config)# router bgp 1000 switch(config-router)# enhanced-error</pre>	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

```

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
<pre> bestpath [always-compare-med as-pathmultipath-relax compare-routerid cost-community ignore igp-metric ignore med {confed missing-as-worst non-deterministic}] </pre> <p>Example:</p> <pre> switch(config-router)# bestpath always-compare-med </pre>	<p>Modifies the best-path algorithm. The optional parameters are as follows:</p> <ul style="list-style-type: none"> • always-compare-med —Compares MED on paths from different autonomous systems. • as-path multipath-relax —Allows load sharing across the providers with different (but equal-length) AS paths. Without this option, the AS paths must be identical for load sharing. • compare-routerid —Compares the router IDs for identical eBGP paths. • cost-community ignore —Ignores the cost community for BGP best-path calculations. • 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 confed —Forces bestpath to do a MED comparison only between paths originated within a confederation. • med missing-as-worst —Treats a missing MED as the highest MED. • med non-deterministic —Does not always pick the best MED path from among the paths from the same autonomous system.
<pre> enforce-first-as </pre> <p>Example:</p> <pre> switch(config-router)# enforce-first-as </pre>	<p>Enforces the neighbor autonomous system to be the first AS number listed in the AS_path attribute for eBGP.</p>

Command	Purpose
<p>log-neighbor-changes</p> <p>Example:</p> <pre>switch(config-router)# log-neighbor-changes</pre>	<p>Generates a system message when any neighbor changes state.</p> <p>Note To suppress neighbor status change messages for a specific neighbor, you can use the log-neighbor-changes disable command in router address-family configuration mode.</p>
<p>router-id <i>id</i></p> <p>Example:</p> <pre>switch(config-router)# router-id 10.165.20.1</pre>	<p>Manually configures the router ID for this BGP speaker.</p>
<p>timers [<i>prefix-peer-wait</i> <i>bgp holdtime</i> prefix-peer-timeout <i>timeout</i> bestpath-limit <i>bestpath-timeout</i>]</p> <p>Example:</p> <pre>switch(config-router)# timers bestpath-limit 300</pre>	<p>Sets BGP timer values. The optional parameters are as follows:</p> <ul style="list-style-type: none"> • <i>prefix-peer-wait</i> —Wait timer for a prefix peer. The range is from 0 to 1200 seconds. The default value is 90. • <i>bgp</i> —BGP session keepalive time. The range is from 0 to 3600 seconds. The default value is 60. • <i>holdtime</i> —Different bgp keepalive and holdtimes. The range is from 0 to 3600 seconds. The default value is 60. • <i>timeout</i> —Prefix peer timeout value. The range is from 0 to 1200 seconds. The default value is 30. • <i>bestpath-timeout</i> —Bestpath timeout in seconds. The default value is 300. When a high-scale BGP setup is expected, the timeout value needs to be set between 480 and 1200, based on the scale. <p>You must manually reset the BGP sessions after configuring this command.</p>

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

Command	Purpose
<p>distance <i>ebgp-distance ibgp-distance local-distance</i></p> <p>Example:</p> <pre>switch(config-router-af)# distance 20 100 200</pre>	<p>Sets the administrative distance for BGP. The range is from 1 to 255. The defaults are as follows:</p> <ul style="list-style-type: none"> • <i>ebgp-distance</i> —20. • <i>ibgp-distance</i> —200. • <i>local-distance</i> —220. Local-distance is the administrative distance used for aggregate discard routes when they are installed in the RIB. <p>After you enter the value for the external administrative distance, you must enter the value for the administrative distance for the internal routes or/and the value for the administrative distance for the local routes depending on your requirement; so that the internal/local routes are also considered in the route administration.</p>
<p>log-neighbor-changes [disable]</p> <p>Example:</p> <pre>switch(config-router-af)# log-neighbor-changes disable</pre>	<p>Generates a system message when this specific neighbor changes state.</p> <p>The disable option suppresses neighbor status changes messages for this specific neighbor.</p>

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

Command	Purpose
<p>description <i>string</i></p> <p>Example:</p> <pre>switch(config-router-neighbor)# description main site</pre>	<p>Sets a descriptive string for this BGP peer. The string can be up to 80 alphanumeric characters.</p>
<p>low-memory exempt</p> <p>Example:</p> <pre>switch(config-router-neighbor)# low-memory exempt</pre>	<p>Exempts this BGP neighbor from a possible shutdown due to a low memory condition.</p>
<p>transport connection-mode passive</p> <p>Example:</p> <pre>switch(config-router-neighbor)# transport connection-mode passive</pre>	<p>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.</p>

Command	Purpose
<p>[no default] remove-private-as [all replace-as]</p> <p>Example:</p> <pre>switch(config-router-neighbor) # remove-private-as</pre>	<p>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.</p> <p>The optional parameters are as follows:</p> <ul style="list-style-type: none"> • no —Disables the command. • default —Moves the command to its default mode. • all —Removes all private-as numbers from the AS-path value. • replace-as —Replaces all private AS numbers with the replace-as AS-path value. <p>See the Guidelines and Limitations for Advanced BGP, on page 14 section for additional information on this command.</p>
<p>update-source <i>interface-type number</i></p> <p>Example:</p> <pre>switch(config-router-neighbor) # update-source ethernet 2/1</pre>	<p>Configures the BGP speaker to use the source IP address of the configured interface for BGP sessions to the peer. This command triggers an automatic notification and session reset for the BGP neighbor sessions. Single-hop iBGP peers support fast external fallover when update-source is configured.</p>

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

Command	Purpose
<p>allows in</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # allows in</pre>	<p>Allows routes that have their own AS in the AS path to be installed in the BRIB.</p>
<p>default-originate [route-map <i>map-name</i>]</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # default-originate</pre>	<p>Generates a default route to the BGP peer.</p>
<p>disable-peer-as-check</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # disable-peer-as-check</pre>	<p>Disables peer AS-number checking while the device advertises routes learned from one node to another node in the same AS path.</p>

Command	Purpose
<p>filter-list <i>list-name</i> {in out}</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # filter-list BGPFilter in</pre>	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.
<p>prefix-list <i>list-name</i> {in out}</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # prefix-list PrefixFilter in</pre>	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.
<p>send-community</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # send-community</pre>	Sends the community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
<p>send-community extended</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # send-community extended</pre>	Sends the extended community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
<p>suppress-inactive</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # suppress-inactive</pre>	Advertises the best (active) routes only to the BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
<p>[no default] as-override</p> <p>Example:</p> <pre>switch(config-router-neighbor-af) # as-override</pre>	<p>no - (Optional) Disables the command.</p> <p>default - (Optional) Moves the command to its default mode.</p> <p>as-override - While sending updates to eBGP peer, replaces in the <i>path</i> attribute all occurrences of the peer's AS number with the local AS number.</p>

Configuring Policy-Based Administrative Distance

You can configure a distance for external BGP (eBGP) and internal BGP (iBGP) routes that match a policy described in the configured route map. The distance configured in the route map is downloaded to the unicast RIB along with the matching routes. BGP uses the best path to determine the administrative distance when downloading next hops in the unicast RIB table. If there is no match or a deny clause in the policy, BGP uses the distance configured in the distance command or the default distance for routes.

The policy-based administrative distance feature is useful when there are two or more different routes to the same destination from two different routing protocols.

Before you begin

You must enable BGP.

SUMMARY STEPS

1. switch# **configure terminal**
2. switch(config)# **ip prefix-list** *name seq number permit prefix-length*
3. switch(config)# **route-map** *map-tag permit sequence-number*
4. switch(config-route-map)# **match ip address prefix-list** *prefix-list-name*
5. switch(config-route-map)# **set distance** *value1 value2 value3*
6. switch(config-route-map)# **exit**
7. switch(config)# **router bgp** *as-number*
8. switch(config-router)# **address-family** {*ipv4 | ipv6 | vpnv4 | vpnv6*} **unicast**
9. switch(config-router-af)# **table-map** *map-name*
10. (Optional) switch(config-router-af)# **show forwarding distribution**
11. (Optional) switch(config)# **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	switch# configure terminal	Enters global configuration mode.
Step 2	switch(config)# ip prefix-list <i>name seq number permit prefix-length</i>	Creates a prefix list to match IP packets or routes with the permit keyword.
Step 3	switch(config)# route-map <i>map-tag permit sequence-number</i>	Creates a route map and enters route-map configuration mode with the permit keyword. If the match criteria for the route is met in the policy, the packet is policy routed.
Step 4	switch(config-route-map)# match ip address prefix-list <i>prefix-list-name</i>	Matches IPv4 network routes based on a prefix list. The prefix-list name can be any alphanumeric string up to 63 characters.
Step 5	switch(config-route-map)# set distance <i>value1 value2 value3</i>	Specifies the administrative distance for interior BGP (iBGP) or exterior BGP (eBGP) routes and BGP routes originated in the local autonomous system. The range is from 1 to 255. After you enter the value for the external administrative distance, you must enter the value for the administrative distance for the internal routes or/and the value for the administrative distance for the local routes depending on your requirement; so that the internal/local routes are also considered in the route administration.
Step 6	switch(config-route-map)# exit	Exits route-map configuration mode.
Step 7	switch(config)# router bgp <i>as-number</i>	Enters BGP mode and assigns the AS number to the local BGP speaker.

	Command or Action	Purpose
Step 8	switch(config-router)# address-family { ipv4 ipv6 vpnv4 vpnv6 } unicast	Enters address family configuration mode.
Step 9	switch(config-router-af)# table-map <i>map-name</i>	Configures the selective administrative distance for a route map for BGP routes before forwarding them to the RIB table. The table-map name can be any alphanumeric string up to 63 characters. Note You can also configure the table-map command under the VRF address-family configuration mode.
Step 10	(Optional) switch(config-router-af)# show forwarding distribution	Displays forwarding information distribution.
Step 11	(Optional) switch(config)# copy running-config startup-config	Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration.

Configuring Multiprotocol BGP

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

Before you begin

You must enable BGP.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	router bgp <i>as-number</i> Example: switch(config)# router bgp 65535 switch(config-router)#	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

	Command or Action	Purpose
Step 3	neighbor <i>ip-address remote-as as-number</i> Example: <pre>switch(config-router)# neighbor 192.168.1.2 remote-as 65534 switch(config-router-neighbor)#</pre>	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 4	address-family { <i>ipv4 ipv6</i> } { <i>unicast multicast</i> } Example: <pre>switch(config-router-neighbor)# address-family ipv4 multicast switch(config-router-neighbor-af)#</pre>	Enters address family configuration mode.
Step 5	(Optional) copy running-config startup-config Example: <pre>switch(config-router-neighbor-af)# copy running-config startup-config</pre>	Saves this configuration change.

Example

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

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

Beginning with Cisco NX-OS Release 7.0(3)I5(2), you can configure BMP on the device.

Before you begin

You must enable BGP (see the [Enabling BGP](#) section).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp as-number**
3. **bmp server server-number**
4. **address ip-address port-number port-number**
5. **description string**

6. **initial-refresh** { *skip* / *delay time* }
7. **initial-delay** *time*
8. **stats-reporting-period** *time*
9. **shutdown**
10. **vrf** *vrf-name*
11. **update-source** <*interface-name*>
12. **neighbor ip-address**
13. **remote-as** *as-number*
14. **bmp-activate-server** *server-number*
15. (Optional) **show bgp bmp server** [*server-number*] [*detail*]
16. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: switch# configure terminal	Enters global configuration mode.
Step 2	router bgp as-number Example: switch(config)# router bgp 200	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)# bmp-server 1	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-bmp)# 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-bmp)# 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-bmp)# 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.

	Command or Action	Purpose
Step 7	initial-delay <i>time</i> Example: <pre>switch(config-router-bmp)# initial-delay 120</pre>	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 <i>time</i> Example: <pre>switch(config-router-bmp)# stats-reporting-period 50</pre>	Configures the time interval in which the BMP server receives the statistics report from BGP neighbors. The range is from 30 to 720 seconds, and the default is disabled.
Step 9	shutdown Example: <pre>switch(config-router-bmp)# shutdown</pre>	Disables the connection to the BMP server.
Step 10	vrf <i>vrf-name</i> Example: <pre>switch(config-router-bmp)# vrf BMP</pre>	Selects vrf in which BMP server is reachable.
Step 11	update-source <i><interface-name></i> Example: <pre>switch(config-router-bmp)# update-source ethernet4/2</pre>	Selects local interface to be used for establishing BMP server connection.
Step 12	neighbor ip-address Example: <pre>switch(config-router-bmp)# neighbor 192.168.1.2</pre>	Enters neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 13	remote-as <i>as-number</i> Example: <pre>switch(config-router-neighbor)# remote-as 65535</pre>	Configures the AS number for a remote BGP peer.
Step 14	bmp-activate-server <i>server-number</i> Example: <pre>switch(config-router-neighbor)# bmp-activate-server 1</pre>	Configures the BMP server to which a neighbor's information should be sent.
Step 15	(Optional) show bgp bmp server [<i>server-number</i>] [<i>detail</i>] Example: <pre>switch(config-router-neighbor)# show bgp bmp server</pre>	Displays BMP server information.
Step 16	(Optional) copy running-config startup-config Example: <pre>switch(config-router-neighbor)# copy running-config startup-config</pre>	Saves this configuration change.

BGP Local Route Leaking

About BGP Local Route Leaking

Beginning with release 9.3(1), NX-OS BGP supports leaking imported VPN routes between:

- The VPN route table and default VRF route table
- The VPN route table and VRF-lite route table
- Border leaf (BL) switch route tables for leaf-to-leaf connectivity

This feature enables the propagation of routes between the route tables. You can control route leaking for a VRF by configuring an import or export map, which now includes an option to allow or prevent incoming locally originated routes and specify whether they should be advertised. Local route leaking is bidirectional, so routes that are locally originated can be leaked from a VRF out to a BGP VPN, and routes that are imported from the BGP VPN can be leaked into a VRF.



Note NX-OS supports a similar feature called centralized route leaking. For information, see [Configuring Layer 3 Virtualization](#).

Guidelines and Limitations for BGP Local Route Leaking

The following are the guidelines and limitations for the BGP local route leaking feature:

- The following Cisco hardware supports this feature:
 - Cisco Nexus 9332C, 9364C, 9300-EX, 9300-FX/FXP/FX2/FX3, and 9300-GX platform switches, and Cisco Nexus 9500 platform switches with 9700-EX/FX line cards
 - Cisco Nexus 9500 platform switches with -R line cards
- When using route-targets, the same route-targets might have duplicated paths pointing to the same remote path, which can negatively impact the switch's memory and performance. Be careful when using route targets.
- Be careful when using local route leaking in a leaf-to-leaf case, where border-leaf routers (BLs) are leaking between the same VRFs. This scenario is more prone to routing loops. We recommend using inbound route-maps to exclude the imported routes from other BLs.
- After a remote path gets withdrawn, it can take up to 20 seconds more for BGP to completely clean up the path.

Configuring Routes Imported from a VPN to Leak into the Default VRF

You can configure a VRF to allow routes that are imported from a BGP VPN to be exported to the default VRF. Use this procedure for a non-default VRF.

Before you begin

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

SUMMARY STEPS

1. **config terminal**
2. **vrf context** *vrf-name*
3. **address-family** *address-family sub family*
4. **export vrf default** [*prefix-limit*] **maproute-map allow-vpn**

DETAILED STEPS

	Command or Action	Purpose
Step 1	config terminal Example: <pre>switch-1# config terminal Enter configuration commands, one per line. End with CNTL/Z. switch-1(config)#</pre>	Enters global configuration mode.
Step 2	vrf context <i>vrf-name</i> Example: <pre>switch-1(config)# vrf context vpn1 switch-1(config-vrf)#</pre>	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	address-family <i>address-family sub family</i> Example: <pre>switch-1(config-vrf)# address-family ipv4 unicast switch-1(config-vrf-af-ipv4)#</pre>	
Step 4	export vrf default [<i>prefix-limit</i>] maproute-map allow-vpn Example: <pre>switch-1(config-vrf-af-ipv4)# export vrf default map vpnmap1 allow-vpn switch-1(config-vrf-af-ipv4)#</pre>	Configures the current VRF to allow routes that are imported from a BGP VPN to be exported to the default VRF.

Configuring Routes Leaked from the Default-VRF to Export to a VPN

You can configure a VRF to allow routes leaked from the default VRF to be exported to a BGP VPN. Use this procedure for a non-default VRF.

Before you begin

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

SUMMARY STEPS

1. **config terminal**
2. **vrf context** *vrf-name*

3. **address-family** *address-family sub family*
4. **import vrf default** [*prefix-limit*] **maproute-map advertise-vpn**

DETAILED STEPS

	Command or Action	Purpose
Step 1	config terminal Example: <pre>switch-1# config terminal Enter configuration commands, one per line. End with CNTL/Z. switch-1(config)#</pre>	Enters global configuration mode.
Step 2	vrf context <i>vrf-name</i> Example: <pre>switch-1(config)# vrf context vpn1 switch-1(config-vrf)#</pre>	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	address-family <i>address-family sub family</i> Example: <pre>switch-1(config-vrf)# address-family ipv4 unicast switch-1(config-vrf-af-ipv4)#</pre>	
Step 4	import vrf default [<i>prefix-limit</i>] maproute-map advertise-vpn Example: <pre>switch-1(config-vrf-af-ipv4)# import vrf map vpnmap1 advertise-vpn switch-1(config-vrf-af-ipv4)#</pre>	Configures the current VRF to allow routes imported from the default VRF to be exported to a BGP VPN.

Configuring Routes Imported from a VPN to Export to a VRF

You can configure a VRF to allow VPN imported routes to be exported to another VRF. Use this procedure for non-default VRFs.

Before you begin

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

SUMMARY STEPS

1. **config terminal**
2. **vrf context** *vrf-name*
3. **address-family** *address-family sub family*
4. **export vrf allow-vpn**

DETAILED STEPS

	Command or Action	Purpose
Step 1	config terminal Example: <pre>switch-1# config terminal Enter configuration commands, one per line. End with CNTL/Z. switch-1(config)#</pre>	Enters global configuration mode.
Step 2	vrf context vrf-name Example: <pre>switch-1(config)# vrf context vpn1 switch-1(config-vrf)#</pre>	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	address-family address-family sub family Example: <pre>switch-1(config-vrf)# address-family ipv4 unicast switch-1(config-vrf-af-ipv4)#</pre>	
Step 4	export vrf allow-vpn Example: <pre>switch-1(config-vrf-af-ipv4)# export vrf allow-vpn nxosv2(config-vrf-af-ipv4)#</pre>	Configures a VRF to allow routes imported from a BGP VPM to be exported to a non-default VRF.

Configuring Routes Imported from a VRF to Export to a VPN

You can configure a VRF to allow routes imported from another VRF to be exported to a BGP VPN. Use this procedure for non-default VRFs.

Before you begin

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

SUMMARY STEPS

1. **config terminal**
2. **vrf context vrf-name**
3. **address-family address-family sub family**
4. **import vrf advertise-vpn**

DETAILED STEPS

	Command or Action	Purpose
Step 1	config terminal Example: <pre>switch-1# config terminal Enter configuration commands, one per line. End</pre>	Enters global configuration mode.

	Command or Action	Purpose
	with CNTL/Z. switch-1(config)#	
Step 2	vrf context <i>vrf-name</i> Example: switch-1(config)# vrf context vpn1 switch-1(config-vrf)#	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	address-family <i>address-family sub family</i> Example: switch-1(config-vrf)# address-family ipv4 unicast switch-1(config-vrf-af-ipv4)#	
Step 4	import vrf advertise-vpn Example: switch-1(config-vrf-af-ipv4)# import vrf advertise-vpn nxosv2(config-vrf-af-ipv4)#	Configures the current VRF to allow routes that are imported from another VRF to be exported to a BGP VPN.

Configuration Examples

The following show sample configurations for the BGP local route leaking feature.

Configuring BGP VPN to Default VRF Reachability

In this example, the configuration enables route re-importation through an intermediate VRF, called VRF_A, which is between the VPN and the default VRF.

```
vrf context VRF_A
  address-family ipv4 unicast
  route-target both auto evpn
  import vrf default map MAP_1 advertise-vpn
  export vrf default map MAP_1 allow-vpn
```

Route re-importation is enabled by using the **advertise-vpn** option to control importing routes from the VPN into VRF_A, and **allow-vpn** for the export map to control exporting VPN-imported routes from VRF_A out to the default VRF. Configuration occurs on the intermediate VRF.

Configuring VPN to VRF-Lite Reachability

In this example, the VPN connects to a tenant VRF, called VRF_A. VRF_A connects a VRF-Lite, called VRF-B. The configuration enables VPN imported routes to be leaked from VRF_A to VRF_B.

```
vrf context VRF_A
  address-family ipv4 unicast
  route-target both auto
  route-target both auto evpn
  route-target import 3:3
  route-target export 2:2
  import vrf advertise-vpn
  export vrf allow-vpn
vrf context VRF_B
  address-family ipv4 unicast
  route-target both 1:1
```

```
route-target import 2:2
route-target export 3:3
```

Route leaking between the two is enabled by using the **allow-vpn** in an export map configured in VRF_A (tenant). The export map in VRF_A allows route imported from the VPN to be leaked into the VRF_B. Routes processed by the export map have the **route-mapexport** and **export-map** attributes added to the route's set of route targets. The import map uses **advertise-vpn** which enables routes that are imported from the VRF-Lite for be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf Reachability

In this example, two VPNs exist and two VRFs exist. VPN_1 is connected to VRF_A and VPN_2 is connected to VRF_B. Both VRFs are route distinguishers (RDs).

```
vrf context VRF_A
  address-family ipv4 unicast
  route-target both auto
  route-target both auto evpn
  route-target import 3:3
  route-target export 2:2
  import vrf advertise-vpn
  export vrf allow-vpn
vrf context VRF_B
  address-family ipv4 unicast
  route-target both 1:1
  route-target import 2:2
  route-target export 3:3
  import vrf advertise-vpn
  export vrf allow-vpn
```

Route leaking between the two is enabled by **allow-vpn** in an export map configured in VRF_A and VRF_B. VPN imported routes have **route-mapexport** and **export-map** attributes added to the route's set of route targets. An import map uses the **advertise-vpn** option which enables routes that are imported from each VRF to be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf with Loop Prevention

In the leaf-to-leaf configuration, you can inadvertently cause loops between the BLs that are leaking between the same VRFs unless you are careful with your route maps:

- You can use an inbound route map in each BL to deny updates from every other BL.
- If a BL originates a route, a standard community can be applied, which enables other BLs to accept the routes. This community is then stripped in the receiving BL.

In the following example, VTEPs 3.3.3.3, 4.4.4.4 and 5.5.5.5 are the BLs.

```
ip prefix-list BL_PREFIX_LIST seq 5 permit 3.3.3.3/32
ip prefix-list BL_PREFIX_LIST seq 10 permit 4.4.4.4/32
ip prefix-list BL_PREFIX_LIST seq 20 permit 5.5.5.5/32
ip community-list standard BL_COMMUNITY seq 10 permit 123:123
route-map INBOUND_MAP permit 5
  match community BL_COMMUNITY
  set community none
route-map INBOUND_MAP deny 10
```

```

    match ip next-hop prefix-list BL_PREFIX_LIST
route-map INBOUND_MAP permit 20
route-map OUTBOUND_SET_COMM permit 10
    match evpn route-type 2 mac-ip
    set community 123:123
route-map SET_COMM permit 10
    set community 123:123
route-map allow permit 10

vrf context vni100
    vni 100
    address-family ipv4 unicast
        route-target import 2:2
        route-target export 1:1
        route-target both auto
        route-target both auto evpn
    import vrf advertise-vpn
    export vrf allow-vpn

vrf context vni200
    vni 200
    address-family ipv4 unicast
        route-target import 1:1
        route-target export 2:2
        route-target both auto
        route-target both auto evpn
    import vrf advertise-vpn
    export vrf allow-vpn

router bgp 100
    template peer rr
        remote-as 100
        update-source loopback0
        address-family l2vpn evpn
            send-community
            send-community extended
            route-map INBOUND_MAP in
            route-map OUTBOUND_SET_COMM out
    neighbor 101.101.101.101
        inherit peer rr
    neighbor 102.102.102.102
        inherit peer rr
    vrf vni100
        address-family ipv4 unicast
            network 3.3.3.100/32 route-map SET_COMM
    vrf vni200
        address-family ipv4 unicast
            network 3.3.3.200/32 route-map SET_COMM

```

In this example, the tenant VRFs for the border leaf (BL) router can leak traffic by enabling extra import export flows, and the route targets in the route maps determine where the routes are imported from or exported to.

Multipath in a VRF

In this example, a VPN has multiple incoming paths. This configuration enables route leaking through an intermediate VRF, called VRF_A, which is between the VPN and another VRF, named VRF_B. Assume that multipathing is enabled in VRF_A.

```

vrf context VRF_A
    address-family ipv4 unicast
        route-target both auto evpn
        route-target export 3:3

```

```

export vrf allow-vpn
vrf context VRF_B
  address-family ipv4 unicast
  route-target import 3:3

```

Route leaking is enabled by **allow-vpn** in the export map configured in VRF_A. When two paths for a given prefix are learnt from a VPN and imported into VRF_A, two different paths exist in VRF_B with the same source RD (VRF_A's local RD). Each route is distinguished by the original source RD (remote RD).

Path Duplication

In this example, the configuration enables a single VPN path to be imported into both VRF_A and VRF_B. Because VRF_A is configured with **export vrf allow-vpn**, VRF_A also leaks its routes into VRF_B. VRF_B then has two paths with same source RD (VRF_A's local RD), each one distinguished by the original source RD (remote RD).

```

vrf context VRF_A
  address-family ipv4 unicast
  route-target import 1:1 evpn
  route-target export 1:1 evpn
  route-target export 2:2
  export vrf allow-vpn
vrf context VRF_B
  address-family ipv4 unicast
  route-target import 1:1 evpn
  route-target import 2:2

```

This configuration creates a situation in which multipathing does not exist.

Displaying BGP Local Route Leaking Information

The following show commands contain information for the BGP local route leaking feature.

Command	Action
show bgp vrf <i>vrf-name</i> process	For a default or non-default VRF, shows the enabled state (Yes or No) of the import advertise-vpn and export allow-vpn options.
show bgp vrf <i>vrf-name</i> ipv4 unicast <i>prefix</i>	Shows information about imported paths, including a list of destinations a route has been imported from.

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 94](#). 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

Context	Affects	Commands
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**).

SUMMARY STEPS

1. **config terminal**
2. **router bgp** *autonomous-system-number*
3. **neighbor** { *ipv4-address|ipv6-address* } **remote-as** *as-number*
4. **graceful-shutdown activate** [*route-map map-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	config terminal Example: <pre>switch-1# configure terminal switch-1(config)#</pre>	Enters global configuration mode.
Step 2	router bgp <i>autonomous-system-number</i> Example: <pre>switch-1(config)# router bgp 110 switch-1(config-router)#</pre>	Enters router configuration mode to create or configure a BGP routing process.

	Command or Action	Purpose
Step 3	<p>neighbor { <i>ipv4-address ipv6-address</i> } remote-as <i>as-number</i></p> <p>Example:</p> <pre>switch-1(config-router)# neighbor 10.0.0.3 remote-as 200 switch-1(config-router-neighbor)#</pre>	Configures the autonomous system (AS) to which the neighbor belongs.
Step 4	<p>graceful-shutdown activate [<i>route-map map-name</i>]</p> <p>Example:</p> <pre>switch-1(config-router-neighbor)# graceful-shutdown activate route-map gshutPeer switch-1(config-router-neighbor)#</pre>	<p>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.</p> <p>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.</p> <p>The devices receiving the gshut community look at the communities of the route and optionally use the communities to apply routing policy.</p>

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 depreffering 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 depreffers routes.

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

SUMMARY STEPS

- configure terminal**
- ip community list standard** *community-list-name seq sequence-number* { **permit** | **deny** } *value*
- route map** *map-tag* { **deny** | **permit** } *sequence-number*
- match community** *community-list-name*
- set local-preference** *local-pref-value*
- exit**
- router bgp** *community-list-name*
- neighbor** { *ipv4-address|ipv6-address* }
- address-family** { *address-family sub family* }
- send community**
- route map** *map-tag in*

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch-1# configure terminal switch-1(config)#</pre>	Enters global configuration mode.
Step 2	ip community list standard <i>community-list-name</i> seq <i>sequence-number</i> { permit deny } <i>value</i> Example: <pre>switch-1(config)# ip community-list standard GSHUT seq 10 permit 65535:0 switch-1(config)#</pre>	Configures a community list and permits or denies routes that have the well-known graceful shutdown community value.
Step 3	route map <i>map-tag</i> {deny permit} <i>sequence-number</i> Example: <pre>switch-1(config)# route-map RM_GSHUT permit 10 switch-1(config-route-map)#</pre>	Configures a route map as sequence 10 and permits routes that have the GRACEFUL_SHUTDOWN community.
Step 4	match community <i>community-list-name</i> Example: <pre>switch-1(config-route-map)# match community GSHUT switch-1(config-route-map)#</pre>	Configures that routes that match the IP community list GSHUT are processed by Route Policy Manager (RPM).
Step 5	set local-preference <i>local-pref-value</i> Example: <pre>switch-1(config-route-map)# set local-preference 10 switch-1(config-route-map)#</pre>	Configures that the routes that match the IP community list GSHUT will be given a specified local preference.
Step 6	exit Example: <pre>switch-1(config-route-map)# exit switch-1(config)#</pre>	Leaves route map configuration and returns to global configuration mode.
Step 7	router bgp <i>community-list-name</i> Example: <pre>switch-1(config)# router bgp 100 switch-1(config-router)#</pre>	Enters router configuration mode and creates a BGP instance.
Step 8	neighbor { <i>ipv4-address</i> <i>ipv6-address</i> } Example: <pre>switch-1(config-router)# neighbor 10.0.0.3 switch-1(config-router-neighbor)#</pre>	Enters route BGP neighbor mode for a specified neighbor.
Step 9	address-family { <i>address-family</i> <i>sub family</i> } Example:	Puts the neighbor into address family (AF) configuration mode.

	Command or Action	Purpose
	<code>nxosv2(config-router-neighbor) # address-family ipv4 unicast nxosv2(config-router-neighbor-af) #</code>	
Step 10	send community Example: <code>nxosv2(config-router-neighbor-af) # send-community nxosv2(config-router-neighbor-af) #</code>	Enables BGP community exchange with the neighbor.
Step 11	route map <i>map-tag</i> in Example: <code>nxosv2(config-router-neighbor-af) # route-map RM_GSHUT in nxosv2(config-router-neighbor-af) #</code>	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**).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp *autonomous-system-number***
3. **graceful-shutdown activate [route-map *map-name*]**

DETAILED STEPS

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

	Command or Action	Purpose
	<pre>switch-1(config-router-neighbor)# graceful-shutdown activate route-map gshutPeer switch-1(config-router-neighbor)#</pre>	<p>GRACEFUL_SHUTDOWN community and applies the route map to the outbound route updates.</p> <p>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.</p> <p>The devices receiving the GRACEFUL_SHUTDOWN community look at the communities of the route and optionally use the communities to apply routing policy.</p>

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**).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system*
3. (Optional) **no graceful-shutdown aware**

DETAILED STEPS

	Command or Action	Purpose
Step 1	<p>configure terminal</p> <p>Example:</p> <pre>switch-1(config)# config terminal switch-1(config)#</pre>	Enters global configuration mode.
Step 2	<p>router bgp <i>autonomous-system</i></p> <p>Example:</p> <pre>switch-1(config)# router bgp 100 switch-1(config-router)#</pre>	Enters router configuration mode and configures a BGP routing process.

	Command or Action	Purpose
Step 3	(Optional) no graceful-shutdown aware Example: <pre>switch-1(config-router)# no graceful-shutdown aware switch-1(config-router)#</pre>	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 deprefering 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 entries in the BGP routing table that have the GRACEFUL_SHUTDOWN community.
show running-config bgp	Shows the running BGP configuration.

Command	Action
show running-config bgp all	Shows all information for the running BGP configuration including information about the graceful shutdown feature.
show bgp <i>address-family</i> neighbors <i>neighbor-address</i>	When the feature is configured for the peer, shows the following: <ul style="list-style-type: none"> • 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. <p>When the graceful-shutdown-activate option is configured in peer context, shows the enabled or disabled state for the feature through <code>graceful-shutdown-active</code>.</p> <p>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:</p> <ul style="list-style-type: none"> • <code>graceful-shutdown-active</code> • <code>graceful-shutdown-aware</code> • <code>graceful-shutdown route-map</code>
show ip bgp <i>address</i>	For the specified address, shows the BGP routing table information, including the following: <ul style="list-style-type: none"> • 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
```

Configuring a Graceful Restart

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



Note Cisco NX-OS Release 10.1(1) supports a higher number of BFD sessions. If BGP sessions are associated with BFD, the BGP **restart-time** may need to be increased to maintain peer connection during ISSU.



Note From the perspective of BGP Graceful Restart, if there are idle peers during a node restart, they can potentially cause traffic loss during an ISSU because they may delay the establishment of the first best-path. It is recommended to either bring all these idle neighbors up, or configure 'shutdown' under each of them, or remove them entirely from the configuration.

Before you begin

You must enable BGP (see the "Enabling BGP" section).

Create the VRFs.

SUMMARY STEPS

1. **configure terminal**

2. **router bgp** *as-number*
3. (Optional) **timers prefix-peer-timeout** *timeout*
4. **graceful-restart**
5. **graceful-restart** {**restart-time** *time*|**stalepath-time** *time*}
6. **graceful-restart-helper**
7. (Optional) **show running-config bgp**
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters configuration mode.
Step 2	router bgp <i>as-number</i> Example: <pre>switch(config)# router bgp 65535 switch(config-router)#</pre>	Creates a new BGP process with the configured autonomous system number.
Step 3	(Optional) timers prefix-peer-timeout <i>timeout</i> Example: <pre>switch(config-router)# timers prefix-peer-timeout 20</pre>	Configures the timeout value (in seconds) for BGP prefix peers. The default value is 90 seconds. Note This command is supported beginning with Cisco NX-OS Release 9.3(3).
Step 4	graceful-restart Example: <pre>switch(config-router)# graceful-restart</pre>	Enables a graceful restart and the graceful restart helper functionality. This command is enabled by default. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 5	graceful-restart { restart-time <i>time</i> stalepath-time <i>time</i> } Example: <pre>switch(config-router)# graceful-restart restart-time 300</pre>	Configures the graceful restart timers. The optional parameters are as follows: <ul style="list-style-type: none"> • restart-time—Maximum time for a restart sent to the BGP peer. The range is from 1 to 3600 seconds. The default is 120. Note Cisco NX-OS Release 10.1(1) supports a higher number of BFD sessions. If BGP sessions are associated with BFD, the BGP restart-time may need to be increased to maintain peer connection during ISSU. • stalepath-time—Maximum time that BGP keeps the stale routes from the restarting BGP peer. The range is from 1 to 3600 seconds. The default is 300.

	Command or Action	Purpose
		In NX-OS software release 10.2(1), a manual reset of a BGP session is needed for the BGP session to advertise Graceful Restart capabilities. For NX-OS software releases 10.2(2) and later, BGP sessions dynamically advertise Graceful Restart capabilities without needing to restart the BGP sessions when this command is enabled.
Step 6	graceful-restart-helper Example: <pre>switch(config-router)# graceful-restart restart-time 300</pre>	With BGP GR disabled, the N9K itself will not necessarily preserve its own forwarding state during certain GR-capable events like SSO, BGP process restart, etc. occurring locally on the N9K. However, as a GR helper, it will support a peer that has advertised its GR capability and is restarting. This means, when the N9K detects the peering has gone down (other than a holdtimer expiration or receipt of a Notification message), the N9K will stale the routes pointing to the peer and will wait for the peer's EOR (or stalepath timeout). When the peer restarts and re-establishes its peering with the N9K, it will re-advertise all its own routes and the N9K will refresh them in its BGP and routing tables. On receipt of the EOR from the peer or the stalepath timeout (whichever occurs first), the N9K will flush any remaining stale routes from that peer. In the absence of helper mode, the N9K would instantly clear out the routes learnt from the remote peer that was restarting which could lead to traffic loss.
Step 7	(Optional) show running-config bgp Example: <pre>switch(config-router)# show running-config bgp</pre>	Displays the BGP configuration.
Step 8	(Optional) copy running-config startup-config Example: <pre>switch(config-router)# copy running-config startup-config</pre>	Saves this configuration change.

Example

This example shows how to enable a graceful restart:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# graceful-restart
switch(config-router)# graceful-restart restart-time 300
switch(config-router)# copy running-config startup-config
```

Configuring Virtualization

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

Before you begin

You must enable BGP.

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose
Step 1	configure terminal Example: <pre>switch# configure terminal switch(config)#</pre>	Enters global configuration mode.
Step 2	vrf context <i>vrf-name</i> Example: <pre>switch(config)# vrf context RemoteOfficeVRF switch(config-vrf)#</pre>	Creates a new VRF and enters VRF configuration mode.
Step 3	exit Example: <pre>switch(config-vrf)# exit switch(config)#</pre>	Exits VRF configuration mode.
Step 4	router bgp <i>as-number</i> Example: <pre>switch(config)# router bgp 65535 switch(config-router)#</pre>	Creates a new BGP process with the configured autonomous system number.
Step 5	vrf <i>vrf-name</i> Example: <pre>switch(config-router)# vrf RemoteOfficeVRF switch(config-router-vrf)#</pre>	Enters the router VRF configuration mode and associates this BGP instance with a VRF.

	Command or Action	Purpose
Step 6	<p>neighbor <i>ip-address</i> remote-as <i>as-number</i></p> <p>Example:</p> <pre>switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65535 switch(config-router--vrf-neighbor)#</pre>	Configures the IP address and AS number for a remote BGP peer.
Step 7	<p>(Optional) copy running-config startup-config</p> <p>Example:</p> <pre>switch(config-router-vrf-neighbor)# copy running-config startup-config</pre>	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
```

Verifying the Advanced BGP Configuration

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

Command	Purpose
show bgp all [summary] [vrf <i>vrf-name</i>]	Displays the BGP information for all address families.
show bgp convergence [vrf <i>vrf-name</i>]	Displays the BGP information for all address families.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] community { regexp <i>expression</i> [community] [no-advertise] [no-export] [no-export-subconfed] } [vrf <i>vrf-name</i>]	Displays the BGP routes that match a BGP community.
show bgp [vrf <i>vrf-name</i>] { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] community-list <i>list-name</i> [vrf <i>vrf-name</i>]	Displays the BGP routes that match a BGP community list.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] extcommunity { regexp <i>expression</i> generic [non-transitive transitive] <i>aa4:nn</i> [exact-match] } [vrf <i>vrf-name</i>]	Displays the BGP routes that match a BGP extended community.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] extcommunity-list <i>list-name</i> [exact-match] [vrf <i>vrf-name</i>]	Displays the BGP routes that match a BGP extended community list.

Command	Purpose
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] extcommunity-list <i>list-name</i> [exact-match] [vrf <i>vrf-name</i>]	Displays the information for BGP route dampening. Use the clear bgp dampening command to clear the route flap dampening information.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] { dampening dampened-paths [regex <i>expression</i>]} [vrf <i>vrf-name</i>]	Displays the BGP route history paths.
show bgp { ipv4 ipv6 vpn4 vpn6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] filter-list <i>list-name</i> [vrf <i>vrf-name</i>]	Displays the information for the BGP filter list.
show bgp { ipv4 ipv6 vpn4 vpn6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] neighbors [<i>ip-address</i> <i>ipv6-prefix</i>] [vrf <i>vrf-name</i>]	Displays the information for BGP peers. Use the clear bgp neighbors command to clear these neighbors.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] { nexthop nexthop-database } [vrf <i>vrf-name</i>]	Displays the information for the BGP route next hop.
show bgp paths	Displays the BGP path information.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] policy <i>name</i> [vrf <i>vrf-name</i>]	Displays the BGP policy information. Use the clear bgp policy command to clear the policy information.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] prefix-list <i>list-name</i> [vrf <i>vrf-name</i>]	Displays the BGP routes that match the prefix list.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] received-paths [vrf <i>vrf-name</i>]	Displays the BGP paths stored for soft reconfiguration.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] regex <i>expression</i> [vrf <i>vrf-name</i>]	Displays the BGP routes that match the AS_path regular expression.
show bgp { ipv4 ipv6 } { unicast multicast } [<i>ip-address</i> <i>ipv6-prefix</i>] route-map <i>map-name</i> [vrf <i>vrf-name</i>]	Displays the BGP routes that match the route map.
show bgp peer-policy <i>name</i> [vrf <i>vrf-name</i>]	Displays the information about BGP peer policies.
show bgp peer-session <i>name</i> [vrf <i>vrf-name</i>]	Displays the information about BGP peer sessions.
show bgp peer-template <i>name</i> [vrf <i>vrf-name</i>]	Displays the information about BGP peer templates. Use the clear bgp peer-template command to clear all neighbors in a peer template.

Command	Purpose
<code>show bgp process</code>	Displays the BGP process information.
<code>show bgp {ipv4 ipv6} unicast neighbors interface</code>	Displays information about BGP peers for the specified interface.
<code>show ip bgp neighbors interface-name</code>	Displays the interface used as a BGP peer.
<code>show ip route ip-address detail vrf all i bw</code>	Displays the link bandwidth EXTCOMM fields. bw:xx (such as bw:40) in the output indicates that BGP peers are sending BGP extended attributes with the bandwidth (for weighted ECMP).
<code>show {ipv4 ipv6} bgp options</code>	Displays the BGP status and configuration information.
<code>show {ipv4 ipv6} mbgp options</code>	Displays the BGP status and configuration information.
<code>show ipv6 routers interface interface</code>	Displays the link-local address of remote IPv6 routers, which is learned through IPv6 ICMP router advertisement.
<code>show running-configuration bgp</code>	Displays the current running BGP configuration.

Monitoring BGP Statistics

To display BGP statistics, use the following commands:

Command	Purpose
<code>show bgp {ipv4 ipv6} {unicast multicast} [ip-address ipv6-prefix] flap-statistics [vrf vrf-name]</code>	Displays the BGP route flap statistics. Use the clear bgp flap-statistics command to clear these statistics.
<code>show bgp {ipv4 ipv6} unicast injected-routes</code>	Displays injected routes in the routing table.
<code>show bgp sessions [vrf vrf-name]</code>	Displays the BGP sessions for all peers. Use the clear bgp sessions command to clear these statistics.
<code>show bgp statistics</code>	Displays the BGP statistics.

Configuration Examples

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
```

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

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

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

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

Related Topics

The following topics can give more information on BGP:

- [Configuring Basic BGP](#)
- [Configuring Route Policy Manager](#)

Additional References

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

MIBs

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

