# Cisco Nexus 3550-T NX-OS Unicast Routing Configuration Guide, Release 10.2(x)

**First Published:** 2022-09-12

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

# CONTENTS

# Preface

This preface includes the following sections:



## Audience

This publication is for network administrators who install, configure, and maintain Cisco Nexus switches.

## Document Conventions

Command descriptions use the following conventions:

| Convention | Description |
|---|---|
| **bold** | Bold text indicates the commands and keywords that you enter literally as shown. |
| *Italic* | Italic text indicates arguments for which you supply the values. |
| [x] | Square brackets enclose an optional element (keyword or argument). |
| [x \| y] | Square brackets enclosing keywords or arguments that are separated by a vertical bar indicate an optional choice. |
| {x \| y} | Braces enclosing keywords or arguments that are separated by a vertical bar indicate a required choice. |
| [x {y \| z}] | Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element. |

| Convention | Description |
|---|---|
| `variable` | Indicates a variable for which you supply values, in context where italics cannot be used. |
| string | A nonquoted set of characters. Do not use quotation marks around the string or the string includes the quotation marks. |

Examples use the following conventions:

| Convention | Description |
|---|---|
| `screen font` | Terminal sessions and information the switch displays are in screen font. |
| **`boldface screen font`** | Information that you must enter is in boldface screen font. |
| *italic screen font* | Arguments for which you supply values are in italic screen font. |
| < > | Nonprinting characters, such as passwords, are in angle brackets. |
| [ ] | Default responses to system prompts are in square brackets. |
| !, # | An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line. |

# Related Documentation for Cisco Nexus 3550-T Switches

The entire Cisco Nexus 3550-T switch documentation set is available at the following URL:

https://www.cisco.com/c/en/us/support/switches/nexus-3550-series/series.html

# Documentation Feedback

To provide technical feedback on this document, or to report an error or omission, please send your comments to nexus9k-docfeedback@cisco.com. We appreciate your feedback.

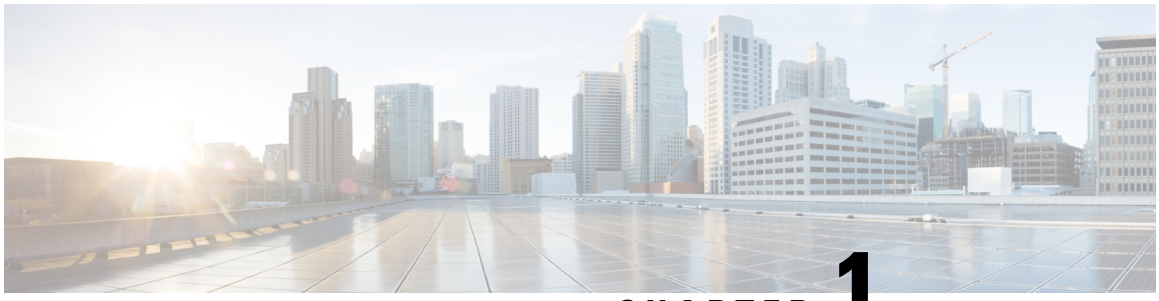
# Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

### Cisco Bug Search Tool

Cisco Bug Search Tool (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.

Preface



CHAPTER 1

# New and Changed Information

This section contains the new and changed information for a release.

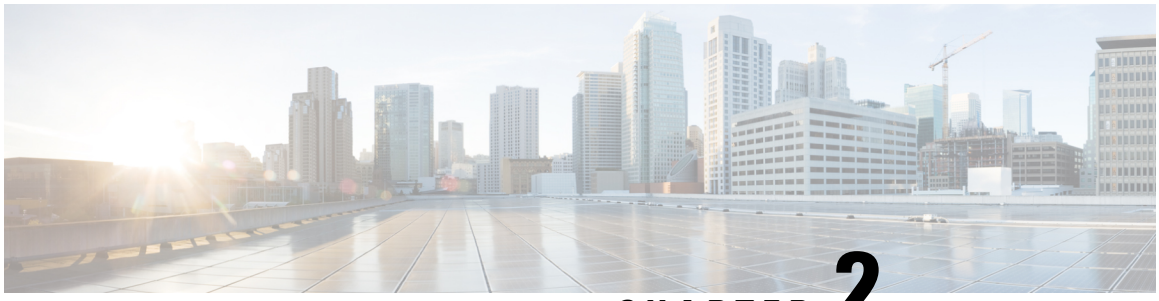


# New and Changed Information

*Table 1: New and Changed Information for Cisco Nexus 3550-T NX-OS Release 10.2(x)*

| Feature | Description | Changed in Release | Where Documented |
|---|---|---|---|
| Support for Layer 3 Virtualization | Cisco NX-OS supports multiple Virtual Routing and Forwarding instances (VRFs). | 10.2(3t) | About Layer 3 Virtualization, on page 171 |
| Updated Guidelines and Limitations for Advanced BGP | Only 48 BGP sessions are validated in Cisco Nexus 3550-T. Support only for IPv4 address family. | 10.2(3t) | Guidelines and Limitations for Advanced BGP, on page 103 |
| Support for Route Policy Manager | The Route Policy Manager provides a route filtering capability in Cisco NX-OS. | 10.2(3t) | Route Policy Manager, on page 13 |

New and Changed Information



CHAPTER 2

# Unicast Routing Overview



## Licensing Requirements

For a complete explanation of Cisco NX-OS licensing recommendations and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

## Information About Layer 3 Unicast Routing

Layer 3 unicast routing involves two basic activities: determining optimal routing paths and packet switching. You can use routing algorithms to calculate the optimal path from the router to a destination. This calculation depends on the algorithm selected, route metrics, and other considerations such as load balancing and alternate path discovery.

### Routing Fundamentals

Routing protocols use a metric to evaluate the best path to the destination. A metric is a standard of measurement, such as a path bandwidth, that routing algorithms use to determine the optimal path to a destination. To aid path determination, routing algorithms initialize and maintain routing tables that contain route information such as the IP destination address, the address of the next router, or the next hop. Destination and next-hop associations tell a router that an IP destination can be reached optimally by sending the packet to a particular router that represents the next hop on the way to the final destination. When a router receives an incoming packet, it checks the destination address and attempts to associate this address with the next hop. See the *Unicast RIB* section for more information about the route table.

Routing tables can contain other information, such as the data about the desirability of a path. Routers compare metrics to determine optimal routes, and these metrics differ depending on the design of the routing algorithm used. See the *Routing Metrics* section.

Routers communicate with one another and maintain their routing tables by transmitting a variety of messages. The routing update message is one such message that consists of all or a portion of a routing table. By analyzing routing updates from all other routers, a router can build a detailed picture of the network topology. A link-state advertisement, which is another example of a message sent between routers, informs other routers of the link state of the sending router. You can also use link information to enable routers to determine optimal routes to network destinations. For more information, see the *Routing Algorithms* section.

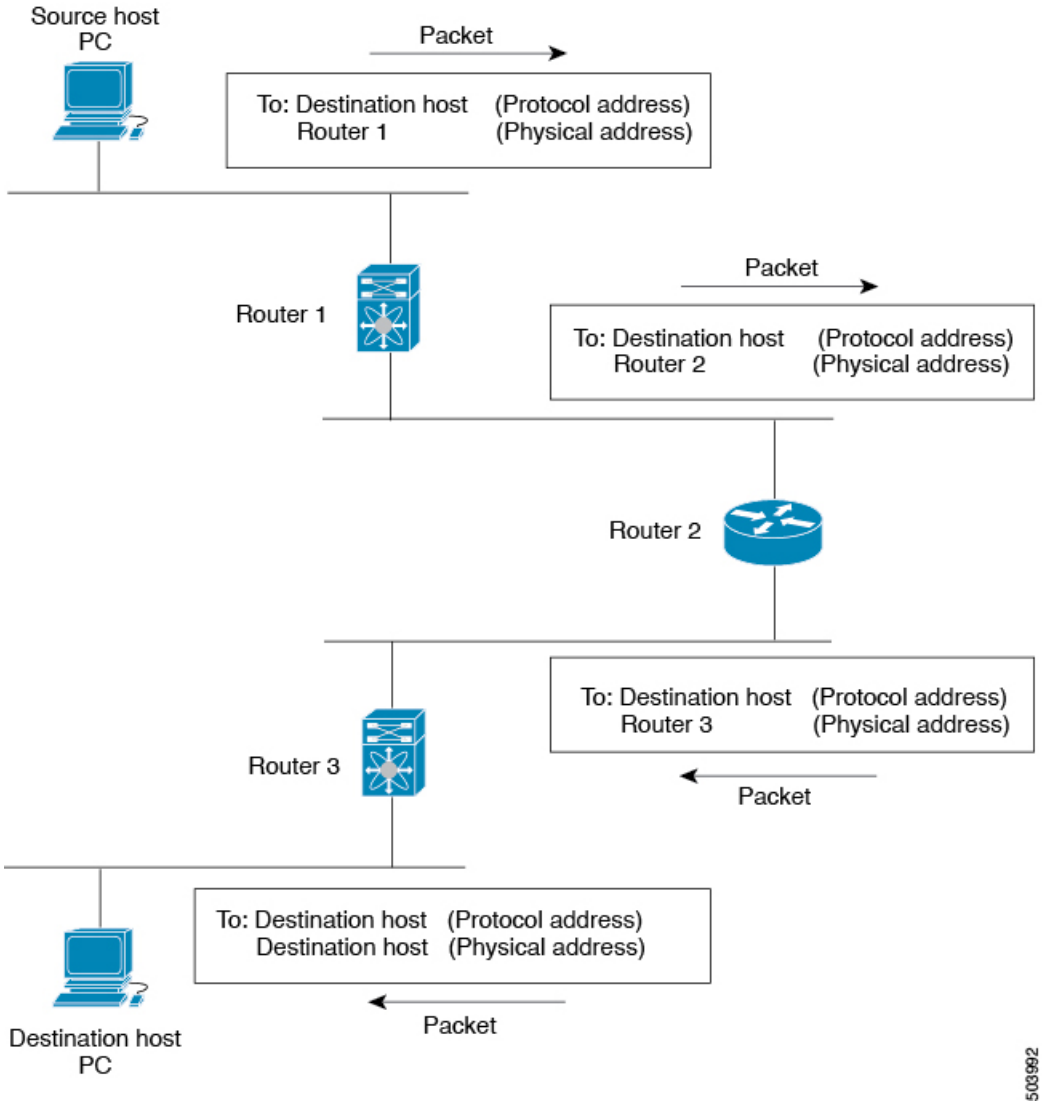
# Packet Switching

In packet switching, a host determines that it must send a packet to another host. Having acquired a router address by some means, the source host sends a packet that is addressed specifically to the router physical (Media Access Control [MAC]-layer) address but with the IP (network layer) address of the destination host.

The router examines the destination IP address and tries to find the IP address in the routing table. If the router does not know how to forward the packet, it typically drops the packet. If the router knows how to forward the packet, it changes the destination MAC address to the MAC address of the next-hop router and transmits the packet.

The next hop might be the ultimate destination host or another router that executes the same switching decision process. As the packet moves through the internetwork, its physical address changes, but its protocol address remains constant (see the following figure).

*Figure 1: Packet Header Updates Through a Network*




# Routing Metrics

Routing algorithms use many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics.

## Path Length

The path length is the most common routing metric. Some routing protocols allow you to assign arbitrary costs to each network link. In this case, the path length is the sum of the costs associated with each link traversed. Other routing protocols define the hop count, which is a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take from a source to a destination.

## Reliability

The reliability, in the context of routing algorithms, is the dependability (in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. The reliability factors that you can take into account when assigning the reliability rating are arbitrary numeric values that you usually assign to network links.

## Routing Delay

The routing delay is the length of time required to move a packet from a source to a destination through the internetwork. The delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, the network congestion on all intermediate network links, and the physical distance that the packet must travel. Because the routing delay is a combination of several important variables, it is a common and useful metric.

## Bandwidth

The bandwidth is the available traffic capacity of a link. For example, a 10-Gigabit Ethernet link is preferable to a 1-Gigabit Ethernet link. Although the bandwidth is the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. For example, if a faster link is busier, the actual time required to send a packet to the destination could be greater.

## Load

The load is the degree to which a network resource, such as a router, is busy. You can calculate the load in a variety of ways, including CPU usage and packets processed per second. Monitoring these parameters on a continual basis can be resource intensive.

## Communication Cost

The communication cost is a measure of the operating cost to route over a link. The communication cost is another important metric, especially if you do not care about performance as much as operating expenditures. For example, the line delay for a private line might be longer than a public line, but you can send packets over your private line rather than through the public lines that cost money for usage time.

# Router IDs

Each routing process has an associated router ID. You can configure the router ID to any interface in the system. If you do not configure the router ID, Cisco NX-OS selects the router ID based on the following criteria:

- Cisco NX-OS prefers loopback0 over any other interface. If loopback0 does not exist, then Cisco NX-OS prefers the first loopback interface over any other interface type.
- If you have not configured a loopback interface, Cisco NX-OS uses the first interface in the configuration file as the router ID. If you configure any loopback interface after Cisco NX-OS selects the router ID, the loopback interface becomes the router ID. If the loopback interface is not loopback0 and you configure loopback0 with an IP address, the router ID changes to the IP address of loopback0.
- If the interface that the router ID is based on changes, that new IP address becomes the router ID. If any other interface changes its IP address, there is no router ID change.

# Convergence

A key aspect to measure for any routing algorithm is how much time a router takes to react to network topology changes. When a part of the network changes for any reason, such as a link failure, the routing information in different routers might not match. Some routers will have updated information about the changed topology, while other routers will still have the old information. The convergence is the amount of time before all routers in the network have updated, matching routing information. The convergence time varies depending on the routing algorithm. Fast convergence minimizes the chance of lost packets caused by inaccurate routing information.

# Route Redistribution

If you have multiple routing protocols configured in your network, you can configure these protocols to share routing information by configuring route redistribution in each protocol. For example, you can configure the Open Shortest Path First (OSPF) protocol to advertise routes learned from the Border Gateway Protocol (BGP). You can also redistribute static routes into any dynamic routing protocol. The router that is redistributing routes from another protocol sets a fixed route metric for those redistributed routes, which prevents incompatible route metrics between the different routing protocols. For example, routes redistributed from EIGRP into OSPF are assigned a fixed link cost metric that OSPF understands.

**Note** You are required to use route maps when you configure the redistribution of routing information.

Route redistribution also uses an administrative distance (see see the *Administrative Distance* section) to distinguish between routes learned from two different routing protocols. The preferred routing protocol is given a lower administrative distance so that its routes are picked over routes from another protocol with a higher administrative distance assigned.

# Administrative Distance

An administrative distance is a rating of the trustworthiness of a routing information source. A higher value indicates a lower trust rating. Typically, a route can be learned through more than one protocol. Administrative distance is used to discriminate between routes learned from more than one protocol. The route with the lowest administrative distance is installed in the IP routing table.

# Stub Routing

You can use stub routing in a hub-and-spoke network topology, where one or more end (stub) networks are connected to a remote router (the spoke) that is connected to one or more distribution routers (the hub). The remote router is adjacent only to one or more distribution routers. The only route for IP traffic to follow into the remote router is through a distribution router. This type of configuration is commonly used in WAN topologies in which the distribution router is directly connected to a WAN. The distribution router can be connected to many more remote routers. Often, the distribution router is connected to 100 or more remote routers. In a hub-and-spoke topology, the remote router must forward all nonlocal traffic to a distribution router, so it becomes unnecessary for the remote router to hold a complete routing table. Generally, the distribution router sends only a default route to the remote router.

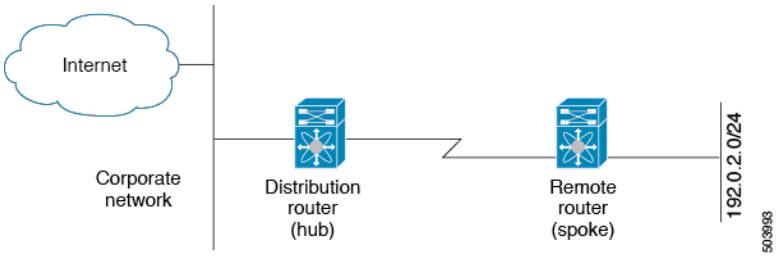
Only specified routes are propagated from the remote (stub) router. The stub router responds to all queries for summaries, connected routes, redistributed static routes, external routes, and internal routes with the

message “inaccessible.” A router that is configured as a stub sends a special peer information packet to all neighboring routers to report its status as a stub router.

Any neighbor that receives a packet that informs it of the stub status does not query the stub router for any routes, and a router that has a stub peer does not query that peer. The stub router depends on the distribution router to send the proper updates to all peers.

The following figure shows a simple hub-and-spoke configuration.

***Figure 2: Simple Hub-and-Spoke Network***




Stub routing does not prevent routes from being advertised to the remote router. The figure **Simple Hub-and-Spoke Network** shows that the remote router can access the corporate network and the Internet through the distribution router only. A full route table on the remote router, in this example, serves no functional purpose because the path to the corporate network and the Internet is always through the distribution router. A larger route table reduces only the amount of memory required by the remote router. The bandwidth and memory used can be lessened by summarizing and filtering routes in the distribution router. In this network topology, the remote router does not need to receive routes that have been learned from other networks because the remote router must send all nonlocal traffic, regardless of its destination, to the distribution router. To configure a true stub network, you should configure the distribution router to send only a default route to the remote router.

OSPF supports stub areas, and the Enhanced Interior Gateway Routing Protocol (EIGRP) supports stub routers.

**Note** The EIGRP stub routing feature should be used only on stub devices. A stub device is defined as a device connected to the network core or distribution layer through which core transit traffic should not flow. The only route for IP traffic to follow into the remote router is through a distribution router. A stub device should not have any EIGRP neighbors other than distribution devices. Ignoring this restriction will cause undesirable behavior.

# Routing Algorithms

Routing algorithms determine how a router gathers and reports reachability information, how it deals with topology changes, and how it determines the optimal route to a destination. Various types of routing algorithms exist, and each algorithm has a different impact on network and router resources. Routing algorithms use a variety of metrics that affect calculation of optimal routes. You can classify routing algorithms by type, such as static or dynamic, and interior or exterior.

# Static Routes and Dynamic Routing Protocols

Static routes are route table entries that you manually configure. These static routes do not change unless you reconfigure them. Static routes are simple to design and work well in environments where network traffic is relatively predictable and where network design is relatively simple.

Because static routing systems cannot react to network changes, you should not use them for large, constantly changing networks. Most routing protocols today use dynamic routing algorithms that adjust to changing network circumstances by analyzing incoming routing update messages. If the message indicates that a network change has occurred, the routing software recalculates routes and sends out new routing update messages. These messages permeate the network, triggering routers to rerun their algorithms and change their routing tables accordingly.

You can supplement dynamic routing algorithms with static routes where appropriate. For example, you should configure each subnetwork with a static route to the IP default gateway or router of last resort (a router to which all unrouteable packets are sent).

# Interior and Exterior Gateway Protocols

You can separate networks into unique routing domains or autonomous systems. An autonomous system is a portion of an internetwork under common administrative authority that is regulated by a particular set of administrative guidelines. Routing protocols that route between autonomous systems are called exterior gateway protocols or interdomain protocols. The Border Gateway Protocol (BGP) is an example of an exterior gateway protocol. Routing protocols used within an autonomous system are called interior gateway protocols or intradomain protocols. EIGRP and OSPF are examples of interior gateway protocols.

# Distance Vector Protocols

Distance vector protocols use distance vector algorithms (also known as Bellman-Ford algorithms) that call for each router to send all or some portion of its routing table to its neighbors. Distance vector algorithms define routes by distance (for example, the number of hops to the destination) and direction (for example, the next-hop router). These routes are then broadcast to the directly connected neighbor routers. Each router uses these updates to verify and update the routing tables.

To prevent routing loops, most distance vector algorithms use split horizon with poison reverse which means that the routes learned from an interface are set as unreachable and advertised back along the interface that they were learned on during the next periodic update. This process prevents the router from seeing its own route updates coming back.

Distance vector algorithms send updates at fixed intervals but can also send updates in response to changes in route metric values. These triggered updates can speed up the route convergence time. The Routing Information Protocol (RIP) is a distance vector protocol.

# Link-State Protocols

The link-state protocols, also known as shortest path first (SPF), share information with neighboring routers. Each router builds a link-state advertisement (LSA) that contains information about each link and directly connected neighbor router.

Each LSA has a sequence number. When a router receives an LSA and updates its link-state database, the LSA is flooded to all adjacent neighbors. If a router receives two LSAs with the same sequence number (from the same router), the router does not flood the last LSA that it received to its neighbors because it wants to

prevent an LSA update loop. Because the router floods the LSAs immediately after it receives them, the convergence time for link-state protocols is minimized.

Discovering neighbors and establishing adjacency is an important part of a link state protocol. Neighbors are discovered using special Hello packets that also serve as keepalive notifications to each neighbor router. Adjacency is the establishment of a common set of operating parameters for the link-state protocol between neighbor routers.

The LSAs received by a router are added to the router's link-state database. Each entry consists of the following parameters:

- Router ID (for the router that originated the LSA)
- Neighbor ID
- Link cost
- Sequence number of the LSA
- Age of the LSA entry

The router runs the SPF algorithm on the link-state database, building the shortest path tree for that router. This SPF tree is used to populate the routing table.

In link-state algorithms, each router builds a picture of the entire network in its routing tables. The link-state algorithms send small updates everywhere, while distance vector algorithms send larger updates only to neighboring routers.

Because they converge more quickly, link-state algorithms are less likely to cause routing loops than distance vector algorithms. However, link-state algorithms require more CPU power and memory than distance vector algorithms and they can be more expensive to implement and support. Link-state protocols are generally more scalable than distance vector protocols.

OSPF is an example of a link-state protocol.

# Layer 3 Virtualization

Cisco NX-OS supports multiple virtual routing and forwarding (VRF) instances and multiple Routing Information Bases (RIBs) to support multiple address domains. Each VRF is associated with a RIB, and this information is collected by the Forwarding Information Base (FIB). A VRF represents a Layer 3 addressing domain. Each Layer 3 interface (logical or physical) belongs to one VRF. For more information, see the *Configuring Layer 3 Virtualization* section.

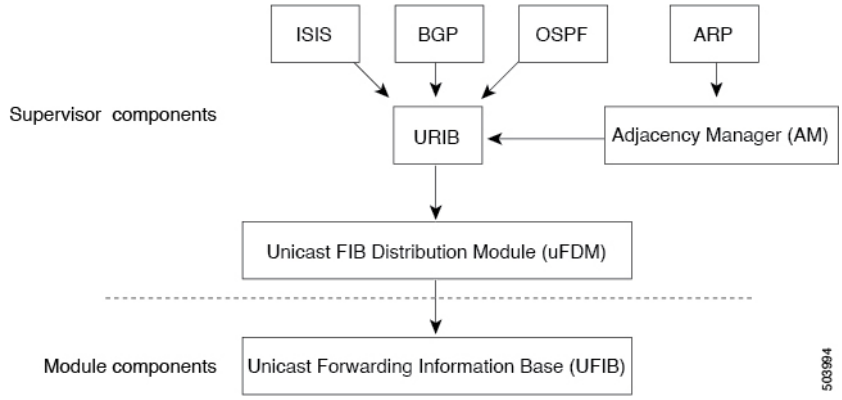
# Cisco NX-OS Forwarding Architecture

The Cisco NX-OS forwarding architecture is responsible for processing all routing updates and populating the forwarding information to all modules in the chassis.

## Unicast RIB

The Cisco NX-OS forwarding architecture consists of multiple components, as shown in the following figure.

*Figure 3: Cisco NX-OS Forwarding Architecture*




The unicast RIB exists on the active supervisor. It maintains the routing table with directly connected routes, static routes, and routes learned from dynamic unicast routing protocols. The unicast RIB also collects adjacency information from sources such as the Address Resolution Protocol (ARP). The unicast RIB determines the best next hop for a given route and populates the FIB by using the services of the unicast FIB Distribution Module (FDM).

Each dynamic routing protocol must update the unicast RIB for any route that has timed out. The unicast RIB then deletes that route and recalculates the best next hop for that route (if an alternate path is available).

# Adjacency Manager

The adjacency manager exists on the active supervisor and maintains adjacency information for different protocols including ARP and static configuration. The most basic adjacency information is the Layer 3 to Layer 2 address mapping discovered by these protocols. Outgoing Layer 2 packets use the adjacency information to complete the Layer 2 header.

The adjacency manager can trigger ARP requests to find a particular Layer 3 to Layer 2 mapping. The new mapping becomes available when the corresponding ARP reply is received and processed.

# Unicast Forwarding Distribution Module

The unicast Forwarding Distribution Module (FDM) exists on the active supervisor and distributes the forwarding path information from the unicast RIB and other sources. The unicast RIB generates forwarding information that the unicast FIB programs into the hardware forwarding tables on the standby supervisor and the modules. The unicast FDM also downloads the FIB information to newly inserted modules.

The unicast FDM gathers adjacency information, rewrite information, and other platform-dependent information when updating routes in the unicast FIB. The adjacency and rewrite information consists of interface, next hop, and Layer 3 to Layer 2 mapping information. The interface and next-hop information is received in route updates from the unicast RIB. The Layer 3 to Layer 2 mapping is received from the adjacency manager.

# FIB

The unicast FIB exists on supervisors and switching modules and builds the information used for the hardware forwarding engine. The unicast FIB receives route updates from the unicast FDM and sends the information

to be programmed in the hardware forwarding engine. The unicast FIB controls the addition, deletion, and modification of routes, paths, and adjacencies.

The unicast FIBs are maintained on a per-VRF and per-address-family basis, that is, one for IPv4 for each configured VRF. Based on route update messages, the unicast FIB maintains a per-VRF prefix and next-hop adjacency information database. The next-hop adjacency data structure contains the next-hop IP address and the Layer 2 rewrite information. Multiple prefixes could share a next-hop adjacency information structure.

## Hardware Forwarding

Cisco NX-OS supports distributed packet forwarding. The ingress port takes relevant information from the packet header and passes the information to the local switching engine. The local switching engine does the Layer 3 lookup and uses this information to rewrite the packet header. The ingress module forwards the packet to the egress port. If the egress port is on a different module, the packet is forwarded using the switch fabric to the egress module. The egress module does not participate in the Layer 3 forwarding decision.

## Software Forwarding

The software forwarding path in Cisco NX-OS is used mainly to handle features that are not supported in the hardware or to handle errors encountered during the hardware processing. Typically, packets with IP options or packets that need fragmentation are passed to the CPU on the active supervisor. All packets that should be switched in the software or terminated go to the supervisor. The supervisor uses the information provided by the unicast RIB and the adjacency manager to make the forwarding decisions. The module is not involved in the software forwarding path.

Software forwarding is controlled by hardware rate limiters.

# Summary of Layer 3 Unicast Routing Features

This section provides a brief introduction to the Layer 3 unicast features and protocols supported in Cisco NX-OS.

## IPv4

Layer 3 uses either the IPv4 protocol. For more information, see the *Configuring IPV4* section.

## OSPF

The Open Shortest Path First (OSPF) protocol is a link-state routing protocol used to exchange network reachability information within an autonomous system. Each OSPF router advertises information about its active links to its neighbor routers. Link information consists of the link type, the link metric, and the neighbor router that is connected to the link. The advertisements that contain this link information are called link-state advertisements. For more information, see the *Configuring OSPFv2* section.

## BGP

The Border Gateway Protocol (BGP) is an inter-autonomous system routing protocol. A BGP router advertises network reachability information to other BGP routers using Transmission Control Protocol (TCP) as its

reliable transport mechanism. The network reachability information includes the destination network prefix, a list of autonomous systems that needs to be traversed to reach the destination, and the next-hop router. Reachability information contains additional path attributes such as preference to a route, origin of the route, community and others. For more information, see the *Configuring Basic BGP* and *Configuring Advanced BGP* sections.

**Note** Cisco NX-OS Release 10.2(3t) supports only IPv4 address family.

## Static Routing

Static routing allows you to enter a fixed route to a destination. This feature is useful for small networks where the topology is simple. Static routing is also used with other routing protocols to control default routes and route distribution. For more information, see the *Configuring Static Routing* section.

## Layer 3 Virtualization

Virtualization allows you to share physical resources across separate management domains. Cisco NX-OS supports Layer 3 virtualization with virtual routing and forwarding (VRF). VRF provides a separate address domain for configuring Layer 3 routing protocols. For more information, see the Configuring Layer 3 Virtualization section.

## Route Policy Manager

The Route Policy Manager provides a route filtering capability in Cisco NX-OS. It uses route maps to filter routes distributed across various routing protocols and between different entities within a given routing protocol. Filtering is based on specific match criteria, which is similar to packet filtering by access control lists.

## First Hop Redundancy Protocols

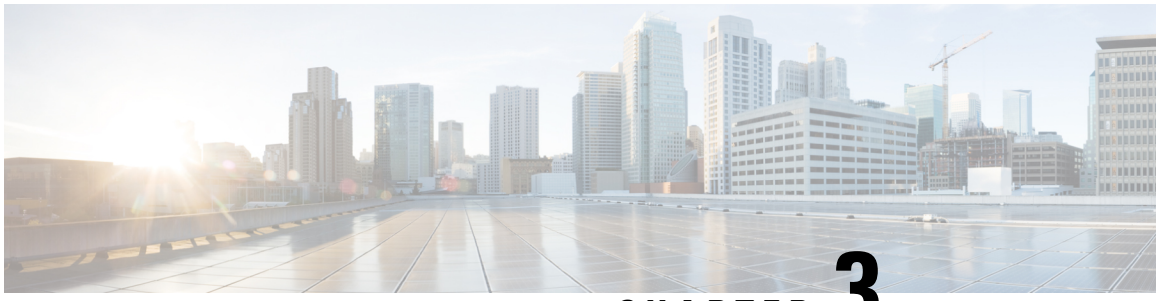
First hop redundancy protocols (FHRP), such as the Virtual Router Redundancy Protocol (VRRP), allow you to provide redundant connections to your hosts. If an active first-hop router fails, the FHRP automatically selects a standby router to take over. You do not need to update the hosts with new IP addresses because the address is virtual and shared between each router in the FHRP group. For more information on VRRP, see the *Configuring VRRP* section.

## Object Tracking

Object tracking allows you to track specific objects on the network, such as the interface line protocol state, IP routing, and route reachability, and take action when the tracked object's state changes. This feature allows you to increase the availability of the network and shorten the recovery time if an object state goes down.

# Related Topics

| Feature Name | Feature Information |
|---|---|
| Layer 3 features | *Cisco Nexus® 3550-T Multicast Routing Configuration* section<br>*Cisco Cisco NX-OS Series NX-OS High Availability and Redundancy Guide*<br>Exploring Autonomous System Numbers: https://www.iana.org/numbers |



CHAPTER 3

# Configuring IPv4

This chapter describes how to configure Internet Protocol version 4 (IPv4), which includes addressing, Address Resolution Protocol (ARP), and Internet Control Message Protocol (ICMP), on the Cisco NX-OS device.

This chapter includes the following sections:



## About IPv4

You can configure IP on the device to assign IP addresses to network interfaces. When you assign IP addresses, you enable the interfaces and allow communication with the hosts on those interfaces.

You can configure an IP address as primary or secondary on a device. An interface can have one primary IP address and multiple secondary addresses. All networking devices on an interface should share the same primary IP address because the packets that are generated by the device always use the primary IPv4 address. Each IPv4 packet is based on the information from a source or destination IP address. For more information, see the Multiple IPv4 Addresses, on page 16 section.

You can use a subnet to mask the IP addresses. A mask is used to determine what subnet an IP address belongs to. An IP address contains the network address and the host address. A mask identifies the bits that denote the network number in an IP address. When you use the mask to subnet a network, the mask is then referred to as a subnet mask. Subnet masks are 32-bit values that allow the recipient of IP packets to distinguish the network ID portion of the IP address from the host ID portion of the IP address.

The IP feature is responsible for handling IPv4 packets that terminate in the supervisor module, as well as forwarding of IPv4 packets, which includes IPv4 unicast/multicast route lookup and software access control list (ACL) forwarding. The IP feature also manages the network interface IP address configuration, duplicate address checks, static routes, and packet send/receive interface for IP clients.

# Multiple IPv4 Addresses

Cisco NX-OS supports multiple IP addresses per interface. You can specify an unlimited number of secondary addresses for a variety of situations. The most common are as follows:

- When there are not enough host IP addresses for a particular network interface. For example, if your subnetting allows up to 254 hosts per logical subnet, but on one physical subnet you must have 300 host addresses, then you can use secondary IP addresses on the routers or access servers to allow you to have two logical subnets that use one physical subnet.
- Two subnets of a single network might otherwise be separated by another network. You can create a single network from subnets that are physically separated by another network by using a secondary address. In these instances, the first network is extended, or layered on top of the second network. A subnet cannot appear on more than one active interface of the router at a time.

**Note** If any device on a network segment uses a secondary IPv4 address, all other devices on that same network interface must also use a secondary address from the same network or subnet. The inconsistent use of secondary addresses on a network segment can quickly cause routing loops.

# LPM Routing Modes

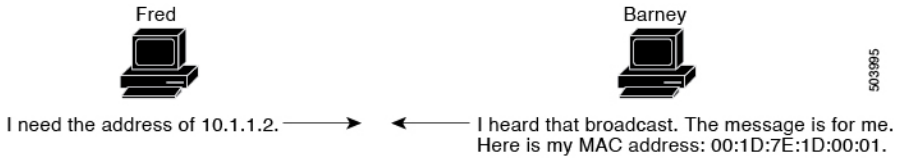
By default, Cisco NX-OS programs routes in a hierarchical fashion to allow for the longest prefix match (LPM) on the device.

# Address Resolution Protocol

Networking devices and Layer 3 switches use Address Resolution Protocol (ARP) to map IP (network layer) addresses to (Media Access Control [MAC]-layer) addresses to enable IP packets to be sent across networks. Before a device sends a packet to another device, it looks in its own ARP cache to see if there is a MAC address and corresponding IP address for the destination device. If there is no entry, the source device sends a broadcast message to every device on the network.

Each device compares the IP address to its own. Only the device with the matching IP address replies to the device that sends the data with a packet that contains the MAC address for the device. The source device adds the destination device MAC address to its ARP table for future reference, creates a data-link header and trailer that encapsulates the packet, and proceeds to transfer the data. The following figure shows the ARP broadcast and response process.

***Figure 4: ARP Process***




When the destination device lies on a remote network that is beyond another device, the process is the same except that the device that sends the data sends an ARP request for the MAC address of the default gateway. After the address is resolved and the default gateway receives the packet, the default gateway broadcasts the

destination IP address over the networks connected to it. The device on the destination device network uses ARP to obtain the MAC address of the destination device and delivers the packet. ARP is enabled by default.

The default system-defined CoPP policy rate limits ARP broadcast packets bound for the supervisor module. The default system-defined CoPP policy prevents an ARP broadcast storm from affecting the control plane traffic but does not affect bridged packets.

# ARP Caching

ARP caching minimizes broadcasts and limits wasteful use of network resources. The mapping of IP addresses to MAC addresses occurs at each hop (device) on the network for every packet sent over an internetwork, which may affect network performance.

ARP caching stores network addresses and the associated data-link addresses in the memory for a period of time, which minimizes the use of valuable network resources to broadcast for the same address each time that a packet is sent. You must maintain the cache entries that are set to expire periodically because the information might become outdated. Every device on a network updates its tables as addresses are broadcast.

# Static and Dynamic Entries in the ARP Cache

Static routing requires that you manually configure the IP addresses, subnet masks, gateways, and corresponding MAC addresses for each interface of each device. Static routing requires more work to maintain the route table. You must update the table each time you add or change routes.

Dynamic routing uses protocols that enable the devices in a network to exchange routing table information with each other. Dynamic routing is more efficient than static routing because the route table is automatically updated unless you add a time limit to the cache. The default time limit is 25 minutes but you can modify the time limit if the network has many routes that are added and deleted from the cache.

# Devices That Do Not Use ARP

When a network is divided into two segments, a bridge joins the segments and filters traffic to each segment based on MAC addresses. The bridge builds its own address table, which uses MAC addresses only. A device has an ARP cache that contains both IP addresses and the corresponding MAC addresses.

Passive hubs are central-connection devices that physically connect other devices in a network. They send messages out on all their ports to the devices and operate at Layer 1 but do not maintain an address table.

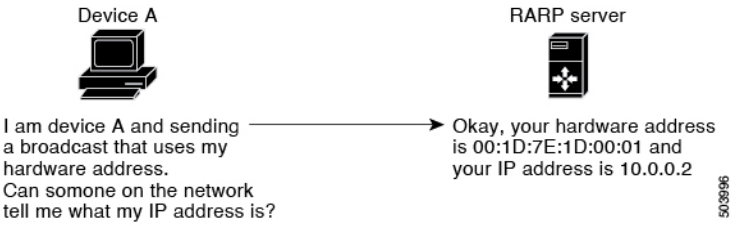
Layer 2 switches determine which port of a device receives a message that is sent only to that port. However, Layer 3 switches are devices that build an ARP cache (table).

# Reverse ARP

Reverse ARP (RARP) as defined by RFC 903 works the same way as ARP, except that the RARP request packet requests an IP address instead of a MAC address. RARP often is used by diskless workstations because this type of device has no way to store IP addresses to use when they boot. The only address that is known is the MAC address because it is burned into the hardware.

Use of RARP requires an RARP server on the same network segment as the router interface. The following figure shows how RARP works.

*Figure 5: Reverse ARP*




RARP has several limitations. Because of these limitations, most businesses use Dynamic Host Control Protocol (DHCP) to assign IP addresses dynamically. DHCP is cost effective and requires less maintenance than RARP. The following are the most important limitations:

- Because RARP uses hardware addresses, if the internetwork is large with many physical networks, a RARP server must be on every segment with an additional server for redundancy. maintaining two servers for every segment is costly.
- Each server must be configured with a table of static mappings between the hardware addresses and IP addresses. Maintenance of the IP addresses is difficult.
- RARP only provides IP addresses of the hosts and not subnet masks or default gateways.

# Proxy ARP

Proxy ARP enables a device that is physically located on one network appear to be logically part of a different physical network connected to the same device or firewall. Proxy ARP allows you to hide a device with a public IP address on a private network behind a router and still have the device appear to be on the public network in front of the router. By hiding its identity, the router accepts responsibility for routing packets to the real destination. Proxy ARP can help devices on a subnet reach remote subnets without configuring routing or a default gateway.

When devices are not in the same data link layer network but in the same IP network, they try to transmit data to each other as if they are on the local network. However, the router that separates the devices does not send a broadcast message because routers do not pass hardware-layer broadcasts and the addresses cannot be resolved.

When you enable proxy ARP on the device and it receives an ARP request, it identifies the request as a request for a system that is not on the local LAN. The device responds as if it is the remote destination for which the broadcast is addressed, with an ARP response that associates the device’s MAC address with the remote destination's IP address. The local device believes that it is directly connected to the destination, while in reality its packets are being forwarded from the local subnetwork toward the destination subnetwork by their local device. By default, proxy ARP is disabled.

# Local Proxy ARP

You can use local proxy ARP to enable a device to respond to ARP requests for IP addresses within a subnet where normally no routing is required. When you enable local proxy ARP, ARP responds to all ARP requests for IP addresses within the subnet and forwards all traffic between hosts in the subnet. Use this feature only on subnets where hosts are intentionally prevented from communicating directly by the configuration on the device to which they are connected.

## Gratuitous ARP

Gratuitous ARP sends a request with an identical source IP address and a destination IP address to detect duplicate IP addresses. Cisco NX-OS supports enabling or disabling gratuitous ARP requests or ARP cache updates.

## ICMP

You can use the Internet Control Message Protocol (ICMP) to provide message packets that report errors and other information that is relevant to IP processing. ICMP generates error messages, such as ICMP destination unreachable messages, ICMP Echo Requests (which send a packet on a round trip between two hosts) and Echo Reply messages. ICMP also provides many diagnostic functions and can send and redirect error packets to the host. By default, ICMP is enabled.

Some of the ICMP message types are as follows:

- Network error messages
- Network congestion messages
- Troubleshooting information
- Timeout announcements

**Note** ICMP redirects are disabled on interfaces where the local proxy ARP feature is enabled.

# Virtualization Support for IPv4

IPv4 supports virtual routing and forwarding (VRF) instances.

# Prerequisites for IPv4

IPv4 has the following prerequisites:

- IPv4 can only be configured on Layer 3 interfaces.

# Guidelines and Limitations for IPv4

IPv4 has the following configuration guidelines and limitations:

**Note** Cisco NX-OS 3550-T series switch does not support ECMP forwarding to achieve lower latency.

- You can configure a secondary IP address only after you configure the primary IP address.

| Parameters | Scale Numbers |
|---|---|
| IP-Host-Route | 4950 ( max) ( per Quad) |
| L3 ARP/Adjacencies | 386 |
| IP-Routes | 2304 ( max) ( per Quad) |

# Default Settings

The table below lists the default settings for IP parameters.

| Parameters | Default |
|---|---|
| ARP timeout | 1500 seconds |
| Proxy ARP | Disabled |

# Configuring IPv4

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

## Configuring IPv4 Addressing

You can assign a primary IP address for a network interface.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | **interface ethernet** *number*<br>**Example:**<br>`switch(config)# interface ethernet 1/3`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **ip address** *ip-address/length* [*secondary*]<br>**Example:**<br>`switch(config-if)# ip address 192.2.1.1 255.0.0.0` | Specifies a primary or secondary IPv4 address for an interface.<br>• The network mask can be a four-part dotted decimal address. For example, 255.0.0.0 indicates that each bit equal to 1 means the corresponding address bit belongs to the network address. |

| | Command or Action | Purpose |
|---|---|---|
| | | • The network mask can be indicated as a slash (/) and a number, which is the prefix length. The prefix length is a decimal value that indicates how many of the high-order contiguous bits of the address comprise the prefix (the network portion of the address). A slash must precede the decimal value and there must be no space between the IP address and the slash. |
| **Step 4** | (Optional) **show ip interface**<br>**Example:**<br>`switch(config-if)# show ip interface` | Displays interfaces configured for IPv4. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring Multiple IP Addresses

You can only add secondary IP addresses after you configure primary IP addresses.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | **interface ethernet** *number*<br>**Example:**<br>`switch(config)# interface ethernet 1/3`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **ip address** *ip-address/length* [*secondary*]<br>**Example:**<br>`switch(config-if)# ip address 192.168.1.1 255.0.0.0 secondary` | Specifies a the configured address as a secondary IPv4 address. |
| **Step 4** | (Optional) **show ip interface**<br>**Example:**<br>`switch(config-if)# show ip interface` | Displays interfaces configured for IPv4. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:** | Saves this configuration change. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-if)# copy running-config startup-config` | **Note** Cisco Nexus® 3550-T switch does not support hardware load balancing across IPv4 paths and installs only first path from an IPv4 ECMP in hardware. The additional paths are only available in software routing table and next one is updated to hardware when first one goes down. |

# Configuring a Static ARP Entry

You can configure a static ARP entry on the device to map IP addresses to MAC hardware addresses, including static multicast MAC addresses.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | **interface ethernet** *number*<br>**Example:**<br>`switch(config)# interface ethernet 1/3`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **ip arp address** *ip-address mac-address*<br>**Example:**<br>`switch(config-if)# ip arp 192.168.1.1 0019.076c.1a78` | Associates an IP address with a MAC address as a static entry. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Saves this configuration change. |

# Configuring Proxy ARP

Configure proxy ARP on the device to determine the media addresses of hosts on other networks or subnets.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | **interface ethernet** *number*<br>**Example:** | Enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# interface ethernet 1/3`<br>`switch(config-if)#` | |
| **Step 3** | **ip proxy-arp**<br>**Example:**<br>`switch(config-if)# ip proxy-arp` | Enables proxy ARP on the interface. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Saves this configuration change. |

# Configuring Local Proxy ARP on Ethernet Interfaces

You can configure local proxy ARP on Ethernet interfaces.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | **interface ethernet** *number*<br>**Example:**<br>`switch(config)# interface ethernet 1/3`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | [**no**]**ip local-proxy-arp**<br>**Example:**<br>`switch(config-if)# ip local-proxy-arp` | Enables Local Proxy ARP on the interface. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Saves this configuration change. |

# Configuring Gratuitous ARP

You can configure gratuitous ARP on an interface.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **interface ethernet** *number*<br>**Example:**<br>`switch(config)# interface ethernet 1/3`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **ip arp gratuitous** {**request** \| **update**]<br>**Example:**<br>`switch(config-if)# ip arp gratuitous request` | Enables gratuitous ARP on the interface. Gratuitous ARP is enabled by default. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Saves this configuration change. |

# Configuring the Interface IP Address for the ICMP Source IP Field

You can configure an interface IP address for the ICMP source IP field to handle ICMP error messages.

**Procedure**

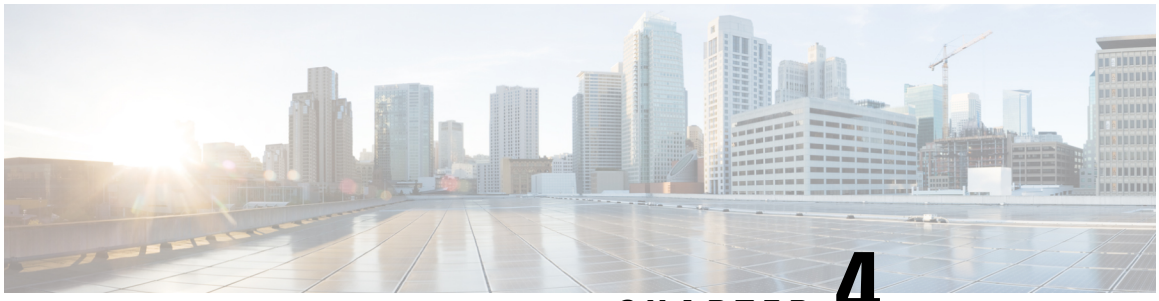
| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | [**no**] **ip source** {**ethernet** *slot/port* \| **loopback** *number* \| **port-channel** *number*} **icmp-errors**<br>**Example:**<br>`switch(config)# ip source loopback 0 icmp-errors` | Configures an interface IP address for the ICMP source IP field to route ICMP error messages. |
| **Step 3** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves this configuration change. |

# Verifying the IPv4 Configuration

To display the IPv4 configuration information, perform one of the following tasks:

| Command | Purpose |
|---|---|
| **show ip adjacency** | Displays the adjacency table. |
| **show ip adjacency summary** | Displays the summary of number of throttle adjacencies. |

| Command | Purpose |
| --- | --- |
| **show ip arp** | Displays the ARP table. |
| **show ip arp summary** | Displays the summary of the number of throttle adjacencies. |
| **show ip interface** | Displays IP-related interface information. |
| **show ip arp statistics** [**vrf** *vrf-name*] | Displays the ARP statistics. |



CHAPTER 4

# Configuring OSPFv2

This chapter describes how to configure Open Shortest Path First version 2 (OSPFv2) for IPv4 networks on the Cisco NX-OS device.

This chapter includes the following sections:



# About OSPFv2

OSPFv2 is an IETF link-state protocol for IPv4 networks. An OSPFv2 router sends a special message, called a hello packet, out each OSPF-enabled interface to discover other OSPFv2 neighbor routers. Once a neighbor is discovered, the two routers compare information in the Hello packet to determine if the routers have compatible configurations. The neighbor routers try to establish adjacency, which means that the routers synchronize their link-state databases to ensure that they have identical OSPFv2 routing information. Adjacent routers share link-state advertisements (LSAs) that include information about the operational state of each link, the cost of the link, and any other neighbor information. The routers then flood these received LSAs out every OSPF-enabled interface so that all OSPFv2 routers eventually have identical link-state databases. When all OSPFv2 routers have identical link-state databases, the network is converged. Each router then uses Dijkstra's Shortest Path First (SPF) algorithm to build its route table.

You can divide OSPFv2 networks into areas. Routers send most LSAs only within one area, which reduces the CPU and memory requirements for an OSPF-enabled router.

OSPFv2 supports IPv4.

**Note** OSPFv2 on Cisco NX-OS supports RFC 2328. This RFC introduced a different method to calculate route summary costs which is not compatible with the calculation used by RFC1583. RFC 2328 also introduced different selection criteria for AS-external paths. It is important_ to ensure that all routers support the same RFC. RFC. Use the **rfc1583compatibility** command if your network includes routers that are only compliant with RFC1583. The default supported RFC standard for OSPFv2 may be different for Cisco NX-OS and Cisco IOS. You must make adjustments to set the values identically. See the OSPF RFC Compatibility Mode Example, on page 74 section for more information.

# Hello Packet

OSPFv2 routers periodically send Hello packets on every OSPF-enabled interface. The hello interval determines how frequently the router sends these Hello packets and is configured per interface. OSPFv2 uses Hello packets for the following tasks:

- Neighbor discovery
- Keepalives
- Bidirectional communications
- Designated router election (see Designated Routers, on page 29)

The Hello packet contains information about the originating OSPFv2 interface and router, including the assigned OSPFv2 cost of the link, the hello interval, and optional capabilities of the originating router. An OSPFv2 interface that receives these Hello packets determines if the settings are compatible with the receiving interface settings. Compatible interfaces are considered neighbors and are added to the neighbor table (see the Neighbors, on page 28 section).

Hello packets also include a list of router IDs for the routers that the originating interface has communicated with. If the receiving interface sees its own router ID in this list, bidirectional communication has been established between the two interfaces.

OSPFv2 uses Hello packets as a keepalive message to determine if a neighbor is still communicating. If a router does not receive a Hello packet by the configured dead interval (usually a multiple of the hello interval), then the neighbor is removed from the local neighbor table.

# Neighbors

An OSPFv2 interface must have a compatible configuration with a remote interface before the two can be considered neighbors. The two OSPFv2 interfaces must match the following criteria:

- Hello interval
- Dead interval
- Area ID (see the Areas, on page 30 section)
- Authentication
- Optional capabilities

If there is a match, the following information is entered into the neighbor table:

- Neighbor ID—The router ID of the neighbor.
- Priority—Priority of the neighbor. The priority is used for designated router election (see the Designated Routers, on page 29 section).
- State—Indication of whether the neighbor has just been heard from, is in the process of setting up bidirectional communications, is sharing the link-state information, or has achieved full adjacency.
- Dead time—Indication of the time since the last Hello packet was received from this neighbor.
- IP Address—The IP address of the neighbor.
- Designated Router—Indication of whether the neighbor has been declared as the designated router or as the backup designated router (see the Designated Routers, on page 29 section).
- Local interface—The local interface that received the Hello packet for this neighbor.

# Adjacency

Not all neighbors establish adjacency. Depending on the network type and designated router establishment, some neighbors become fully adjacent and share LSAs with all their neighbors, while other neighbors do not. For more information, see the Designated Routers, on page 29 section.

Adjacency is established using Database Description (DD) packets, Link State Request (LSR) packets, and Link State Update (LSU) packets in OSPF. The Database Description packet includes just the LSA headers from the link-state database of the neighbor (see the Link-State Advertisements, on page 31 section). The local router compares these headers with its own link-state database and determines which LSAs are new or updated. The local router sends an LSR packet for each LSA that it needs new or updated information on. The neighbor responds with an LSU packet. This exchange continues until both routers have the same link-state information.

# Designated Routers

Networks with multiple routers present a unique situation for OSPF. If every router floods the network with LSAs, the same link-state information is sent from multiple sources. Depending on the type of network, OSPFv2 might use a single router, the designated router (DR), to control the LSA floods and represent the network to the rest of the OSPFv2 area (see the Areas section). If the DR fails, OSPFv2 selects a backup designated router (BDR). If the DR fails, OSPFv2 uses the BDR.

Network types are as follows:

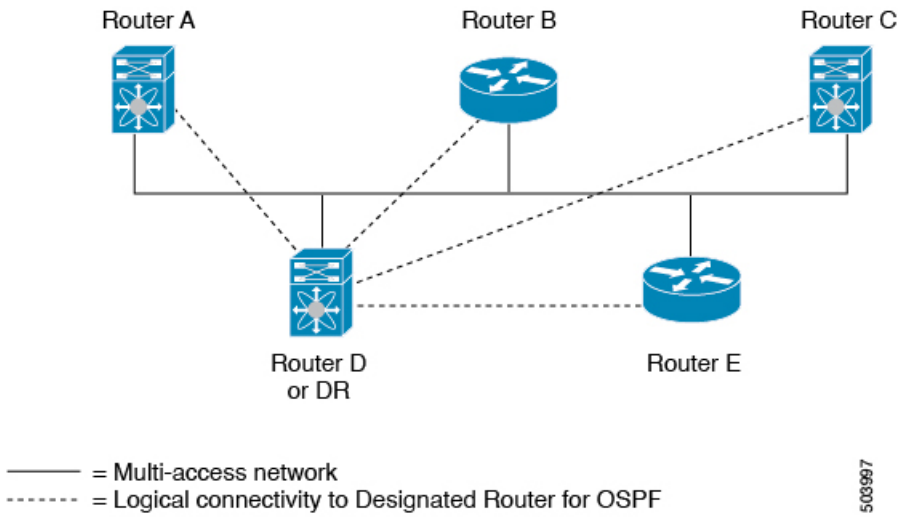
- Point-to-point—A network that exists only between two routers. All neighbors on a point-to-point network establish adjacency and there is no DR.
- Broadcast—A network with multiple routers that can communicate over a shared medium that allows broadcast traffic, such as Ethernet. OSPFv2 routers establish a DR and a BDR that controls LSA flooding on the network. OSPFv2 uses the well-known IPv4 multicast addresses 224.0.0.5 and a MAC address of 0100.5300.0005 to communicate with neighbors.

The DR and BDR are selected based on the information in the Hello packet. When an interface sends a Hello packet, it sets the priority field and the DR and BDR field if it knows who the DR and BDR are. The routers follow an election procedure based on which routers declare themselves in the DR and BDR fields and the priority field in the Hello packet. As a final tie breaker, OSPFv2 chooses the highest router IDs as the DR and BDR.

All other routers establish adjacency with the DR and the BDR and use the IPv4 multicast address 224.0.0.6 to send LSA updates to the DR and BDR. The figure below shows this adjacency relationship between all routers and the DR.

DRs are based on a router interface. A router might be the DR for one network and not for another network on a different interface.

***Figure 6: DR in Multi-Access Network***



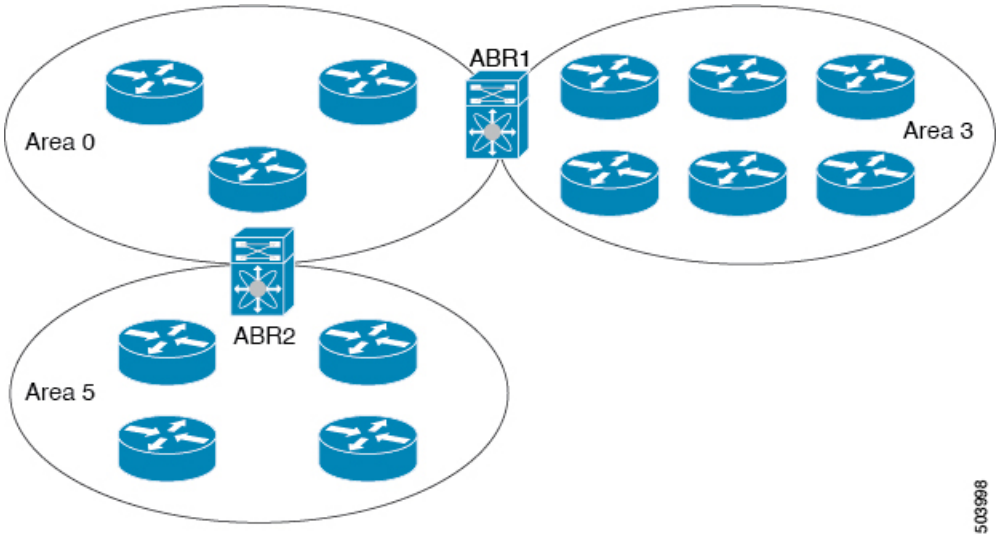

# Areas

You can limit the CPU and memory requirements that OSPFv2 puts on the routers by dividing an OSPFv2 network into areas. An area is a logical division of routers and links within an OSPFv2 domain that creates separate subdomains. LSA flooding is contained within an area, and the link-state database is limited to links within the area. You can assign an area ID to the interfaces within the defined area. The Area ID is a 32-bit value that you can enter as a number or in dotted decimal notation, such as 10.2.3.1.

Cisco NX-OS always displays the area in dotted decimal notation.

If you define more than one area in an OSPFv2 network, you must also define the backbone area, which has the reserved area ID of 0. If you have more than one area, then one or more routers become area border routers (ABRs). The figure shows how an ABR connects to both the backbone area and at least one other defined area.

*Figure 7: OSPFv2 Areas*




The ABR has a separate link-state database for each area to which it connects. The ABR sends Network Summary (type 3) LSAs (see the Route Summarization, on page 36 section) from one connected area to the backbone area. The backbone area sends summarized information about one area to another area. In the OSPFv2 Areas Figure, Area 0 sends summarized information about Area 5 to Area 3.

OSPFv2 defines one other router type: the autonomous system boundary router (ASBR). This router connects an OSPFv2 area to another autonomous system. An autonomous system is a network controlled by a single technical administration entity. OSPFv2 can redistribute its routing information into another autonomous system or receive redistributed routes from another autonomous system. For more information, see the Advanced Features, on page 34 section.

# Link-State Advertisements

OSPFv2 uses link-state advertisements (LSAs) to build its routing table.

## Link-State Advertisement Types

OSPFv2 uses link-state advertisements (LSAs) to build its routing table.

The table shows the LSA types supported by Cisco NX-OS.

*Table 2: Table 5-1 LSA Types*

| Type | Name | Description |
|---|---|---|
| 1 | Router LSA | LSA sent by every router. This LSA includes the state and the cost of all links and a list of all OSPFv2 neighbors on the link. Router LSAs trigger an SPF recalculation. Router LSAs are flooded to local OSPFv2 area. |
| 2 | Network LSA | LSA sent by the DR. This LSA lists all routers in the multi-access network. Network LSAs trigger an SPF recalculation. See the Designated Routers section. |

| Type | Name | Description |
|---|---|---|
| 3 | Network Summary LSA | LSA sent by the area border router to an external area for each destination in the local area. This LSA includes the link cost from the area border router to the local destination. See the Areas section. |
| 4 | ASBR Summary LSA | LSA sent by the area border router to an external area. This LSA advertises the link cost to the ASBR only. See the Areas section. |
| 5 | AS External LSA | LSA generated by the ASBR. This LSA includes the link cost to an external autonomous system destination. AS External LSAs are flooded throughout the autonomous system. See the Areas section. |
| 7 | NSSA External LSA | LSA generated by the ASBR within a not-so-stubby area (NSSA). This LSA includes the link cost to an external autonomous system destination. NSSA External LSAs are flooded only within the local NSSA. See the Areas section. |
| 9–11 | Opaque LSAs | LSA used to extend OSPF. See the Opaque LSAs section. |

## Link Cost

Each OSPFv2 interface is assigned a link cost. The cost is an arbitrary number. By default, Cisco NX-OS assigns a cost that is the configured reference bandwidth divided by the interface bandwidth. By default, the reference bandwidth is 40 Gb/s. The link cost is carried in the LSA updates for each link.

## Flooding and LSA Group Pacing

When an OSPFv2 router receives an LSA, it forwards that LSA out every OSPF-enabled interface, flooding the OSPFv2 area with this information. This LSA flooding guarantees that all routers in the network have identical routing information. LSA flooding depends on the OSPFv2 area configuration (see the Areas, on page 30 section). The LSAs are flooded based on the link-state refresh time (every 30 minutes by default). Each LSA has its own link-state refresh time.

You can control the flooding rate of LSA updates in your network by using the LSA group pacing feature. LSA group pacing can reduce high CPU or buffer usage. This feature groups LSAs with similar link-state refresh times to allow OSPFv2 to pack multiple LSAs into an OSPFv2 Update message.

By default, LSAs with link-state refresh times within 10 seconds of each other are grouped together. You should lower this value for large link-state databases or raise it for smaller databases to optimize the OSPFv2 load on your network.

## Link-State Database

Each router maintains a link-state database for the OSPFv2 network. This database contains all the collected LSAs, and includes information on all the routes through the network. OSPFv2 uses this information to calculate the bast path to each destination and populates the routing table with these best paths.

LSAs are removed from the link-state database if no LSA update has been received within a set interval, called the MaxAge. Routers flood a repeat of the LSA every 30 minutes to prevent accurate link-state information from being aged out. Cisco NX-OS supports the LSA grouping feature to prevent all LSAs from refreshing at the same time. For more information, see the Flooding and LSA Group Pacing, on page 32 section.

## Opaque LSAs

Opaque LSAs allow you to extend OSPF functionality. Opaque LSAs consist of a standard LSA header followed by application-specific information. This information might be used by OSPFv2 or by other applications. OSPFv2 uses Opaque LSAs to support OSPFv2 Graceful Restart capability (see the High Availability and Graceful Restart, on page 37 section). Three Opaque LSA types are defined as follows:

- LSA type 9—Flooded to the local network.
- LSA type 10—Flooded to the local area.
- LSA type 11—Flooded to the local autonomous system.

# OSPFv2 and the Unicast RIB

OSPFv2 runs the Dijkstra shortest path first algorithm on the link-state database. This algorithm selects the best path to each destination based on the sum of all the link costs for each link in the path. The resultant shortest path for each destination is then put in the OSPFv2 route table. When the OSPFv2 network is converged, this route table feeds into the unicast RIB. OSPFv2 communicates with the unicast RIB to do the following:

- Add or remove routes
- Handle route redistribution from other protocols
- Provide convergence updates to remove stale OSPFv2 routes and for stub router advertisements (see the OSPFv2 Stub Router Advertisements section)

OSPFv2 also runs a modified Dijkstra algorithm for fast recalculation for summary and external (type 3, 4, 5, and 7) LSA changes.

# Authentication

You can configure authentication on OSPFv2 messages to prevent unauthorized or invalid routing updates in your network. Cisco NX-OS supports two authentication methods:

- Simple password authentication
- MD5 authentication digest

You can configure the OSPFv2 authentication for an OSPFv2 area or per interface.

## Simple Password Authentication

Simple password authentication uses a simple clear-text password that is sent as part of the OSPFv2 message. The receiving OSPFv2 router must be configured with the same clear-text password to accept the OSPFv2 message as a valid route update. Because the password is in clear text, anyone who can watch traffic on the network can learn the password.

# Cryptographic Authentication

Cryptographic authentication uses an encrypted password for OSPFv2 authentication. The transmitter computes a code using the packet to be transmitted and the key string, inserts the code and the key ID in the packet, and transmits the packet. The receiver validates the code in the packet by computing the code locally using the received packet and the key string (corresponding to the key ID in the packet) configured locally.

Both message digest 5 (MD5) and hash-based message authentication code secure hash algorithm (HMAC-SHA) cryptographic authentication are supported.

## MD5 Authentication

You should use MD5 authentication to authenticate OSPFv2 messages. You configure a password that is shared at the local router and all remote OSPFv2 neighbors. For each OSPFv2 message, Cisco NX-OS creates an MD5 one-way message digest based on the message itself and the encrypted password. The interface sends this digest with the OSPFv2 message. The receiving OSPFv2 neighbor validates the digest using the same encrypted password. If the message has not changed, the digest calculation is identical and the OSPFv2 message is considered valid.

MD5 authentication includes a sequence number with each OSPFv2 message to ensure that no message is replayed in the network.

## HMAC-SHA Authentication

OSPFv2 supports RFC 5709 to allow the use of HMAC-SHA algorithms, which offer more security than MD5. The HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384. and HMAC-SHA-512 algorithms are supported for OSPFv2 authentication.

# Advanced Features

Cisco NX-OS supports advanced OSPFv3 features that enhance the usability and scalability of OSPFv2 in the network.

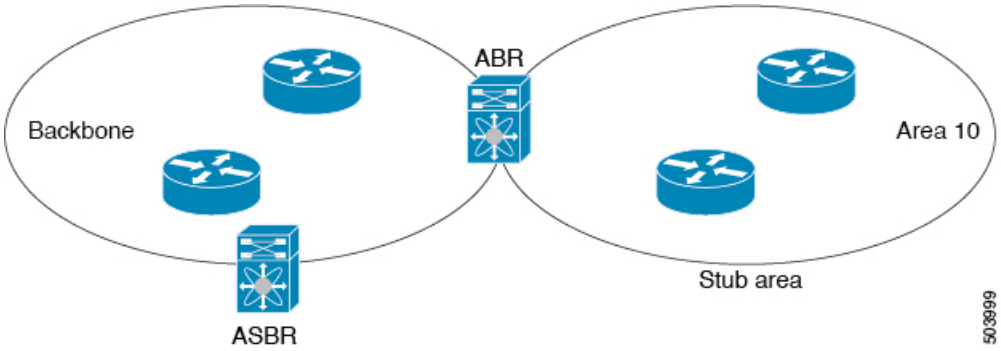
# Stub Area

You can limit the amount of external routing information that floods an area by making it a stub area. A stub area is an area that does not allow AS External (type 5) LSAs (see the *Link State Advertisement* section). These LSAs are usually flooded throughout the local autonomous system to propagate external route information. Stub areas have the following requirements:

- All routers in the stub area are stub routers.
- No ASBR routers exist in the stub area.
- You cannot configure virtual links in the stub area.

The following figure shows an example of an OSPFv2 autonomous system where all routers in area 0.0.0.10 have to go through the ABR to reach external autonomous systems. Area 0.0.0.10 can be configured as a stub area.

*Figure 8: Stub Area*




Stub areas use a default route for all traffic that needs to go through the backbone area to the external autonomous system. The default route is 0.0.0.0 for IPv4.

# Not-So-Stubby Area

A Not-so-Stubby Area (NSSA) is similar to a stub area, except that an NSSA allows you to import autonomous system external routes within an NSSA using redistribution. The NSSA ASBR redistributes these routes and generates NSSA External (type 7) LSAs that it floods throughout the NSSA. You can optionally configure the ABR that connects the NSSA to other areas to translate this NSSA External LSA to AS External (type 5) LSAs. The ABR then floods these AS External LSAs throughout the OSPFv2 autonomous system. Summarization and filtering are supported during the translation. See the Link-State Advertisements, on page 31 section for information about NSSA External LSAs.

You can, for example, use NSSA to simplify administration if you are connecting a central site using OSPFv2 to a remote site that is using a different routing protocol. Before NSSA, the connection between the corporate site border router and a remote router could not be run as an OSPFv2 stub area because routes for the remote site could not be redistributed into a stub area. With NSSA, you can extend OSPFv2 to cover the remote connection by defining the area between the corporate router and remote router as an NSSA.

The backbone Area 0 cannot be an NSSA.

**Note** OSPF is compliant with RFC 3101 section 2.5(3). When an Area Border Router attached to a Not-so-Stubby Area receives a default route LSA with P-bit clear, it should be ignored. OSPF had been previously adding the default route under these conditions.

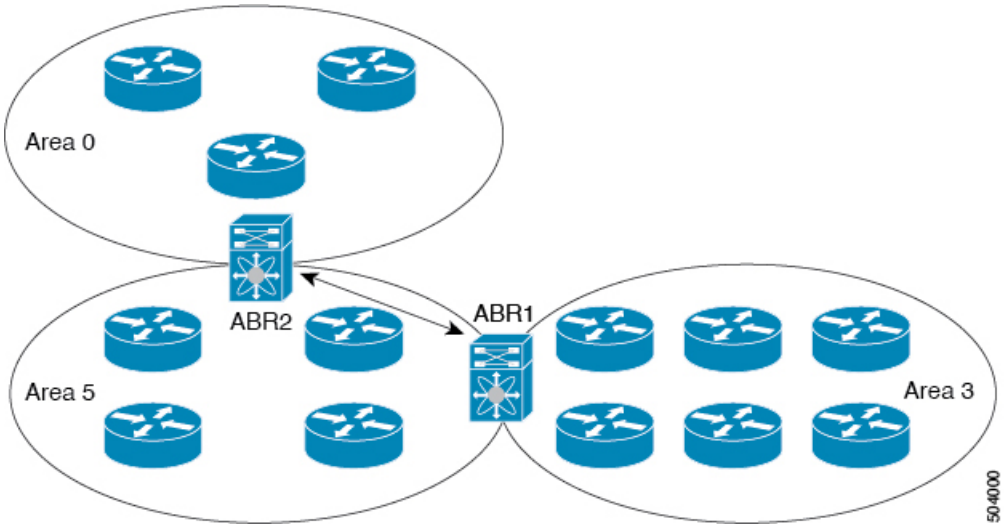
If you have already designed your networks with RFC non-compliant behavior and expect a default route to be added on NSSA ABR, you will see a change in behavior when you upgrade.

If you decide to continue with the old behavior, you have the option to enable it with the **default-route nssa-abr pbit-clear** command.

# Virtual Links

Virtual links allow you to connect an OSPFv2 area ABR to a backbone area ABR when a direct physical connection is not available. The figure shows a virtual link that connects Area 3 to the backbone area through Area 5.

*Figure 9: Virtual Links*




You can also use virtual links to temporarily recover from a partitioned area, which occurs when a link within the area fails, isolating part of the area from reaching the designated ABR to the backbone area.

# Route Redistribution

OSPFv2 can learn routes from other routing protocols by using route redistribution. You configure OSPFv2 to assign a link cost for these redistributed routes or a default link cost for all redistributed routes.

Route redistribution uses route maps to control which external routes are redistributed. You must configure a route map with the redistribution to control which routes are passed into OSPFv2. A route map allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. You can use route maps to modify parameters in the AS External (type 5) and NSSA External (type 7) LSAs before these external routes are advertised in the local OSPFv2 autonomous system. See the Configuring Route Policy Manager section, for information about configuring route maps.

# Route Summarization

Because OSPFv2 shares all learned routes with every OSPF-enabled router, you might want to use route summarization to reduce the number of unique routes that are flooded to every OSPF-enabled router. Route summarization simplifies route tables by replacing more-specific addresses with an address that represents all the specific addresses. For example, you can replace 10.1.1.0/24, 10.1.2.0/24, and 10.1.3.0/24 with one summary address, 10.1.0.0/16.

Typically, you would summarize at the boundaries of area border routers (ABRs). Although you could configure summarization between any two areas, it is better to summarize in the direction of the backbone so that the backbone receives all the aggregate addresses and injects them, already summarized, into other areas. The two types of summarization are as follows

- Inter-area route summarization
- External route summarization

You configure inter-area route summarization on ABRs, summarizing routes between areas in the autonomous system. To take advantage of summarization, you should assign network numbers in areas in a contiguous way to be able to lump these addresses into one range.

External route summarization is specific to external routes that are injected into OSPFv2 using route redistribution. You should make sure that external ranges that are being summarized are contiguous. Summarizing overlapping ranges from two different routers could cause packets to be sent to the wrong destination. Configure external route summarization on ASBRs that are redistributing routes into OSPF.

When you configure a summary address, Cisco NX-OS automatically configures a discard route for the summary address to prevent routing black holes and route loops.

# High Availability and Graceful Restart

Cisco NX-OS provides a multilevel high-availability architecture. OSPFv2 supports stateful restart, which is also referred to as non-stop routing (NSR). If OSPFv2 experiences problems, it attempts to restart from its previous run-time state. The neighbors do not register any neighbor event in this case. If the first restart is not successful and another problem occurs, OSPFv2 attempts a graceful restart.

A graceful restart, or nonstop forwarding (NSF), allows OSPFv2 to remain in the data forwarding path through a process restart. When OSPFv2 needs to perform a graceful restart, it sends a link-local opaque (type 9) LSA, called a grace LSA (see the Opaque LSAs, on page 33 section). This restarting OSPFv2 platform is called NSF capable.

The grace LSA includes a grace period, which is a specified time that the neighbor OSPFv2 interfaces hold onto the LSAs from the restarting OSPFv2 interface. (Typically, OSPFv2 tears down the adjacency and discards all LSAs from a down or restarting OSPFv2 interface.) The participating neighbors, which are called NSF helpers, keep all LSAs that originate from the restarting OSPFv2 interface as if the interface was still adjacent.

When the restarting OSPFv2 interface is operational again, it rediscovers its neighbors, establishes adjacency, and starts sending its LSA updates again. At this point, the NSF helpers recognize that the graceful restart has finished.

Stateful restart is used in the following scenarios:

- First recovery attempt after the process experiences problems

Graceful restart is used in the following scenarios:

- Second recovery attempt after the process experiences problems within a 4-minute interval
- Manual restart of the process using the **restart ospf** command

# OSPFv2 Stub Router Advertisements

You can configure an OSPFv2 interface to act as a stub router using the OSPFv2 Stub Router Advertisements feature. Use this feature when you want to limit the OSPFv2 traffic through this router, such as when you want to introduce a new router to the network in a controlled manner or limit the load on a router that is already overloaded. You might also want to use this feature for various administrative or traffic engineering reasons.

OSPFv2 stub router advertisements do not remove the OSPFv2 router from the network topology, but they do prevent other OSPFv2 routers from using this router to route traffic to other parts of the network. Only the traffic that is destined for this router or directly connected to this router is sent.

OSPFv2 stub router advertisements mark all stub links (directly connected to the local router) to the cost of the local OSPFv2 interface. All remote links are marked with the maximum cost (0xFFFF).

## Multiple OSPFv2 Instances

Cisco Nexus® 3550-T switch supports multiple instances of the OSPFv2 protocol that run on the same node. You cannot configure multiple instances over the same interface. By default, every instance uses the same system router ID. You must manually configure the router ID for each instance if the instances are in the same OSPFv2 autonomous system. For the number of supported OSPFv2 instances, see the *Cisco Nexus® 3550-T Verified Scalability Guide*.

## SPF Optimization

Cisco NX-OS optimizes the SPF algorithm in the following ways:

- Partial SPF for Network (type 2) LSAs, Network Summary (type 3) LSAs, and AS External (type 5) LSAs—When there is a change on any of these LSAs, Cisco NX-OS performs a faster partial calculation rather than running the whole SPF calculation.
- SPF timers—You can configure different timers for controlling SPF calculations. These timers include exponential backoff for subsequent SPF calculations. The exponential backoff limits the CPU load of multiple SPF calculations.

## Virtualization Support for OSPFv2

Cisco Nexus® 3550-T switch supports multiple process instances for OSPFv2. Each OSPF instance can support multiple virtual routing and forwarding (VRF) instances, up to the system limit. For the number of supported OSPFv2 instances, see the *Cisco Nexus® 3550-T Verified Scalability Guide*.

# Prerequisites for OSPFv2

OSPFv2 has the following prerequisites:

- You must be familiar with routing fundamentals to configure OSPF.
- You are logged on to the switch.
- You have configured at least one interface for IPv4 that can communicate with a remote OSPFv2 neighbor.
- You have completed the OSPFv2 network strategy and planning for your network. For example, you must decide whether multiple areas are required.
- You have enabled the OSPF feature (see the Enabling OSPFv2 section).

# Guidelines and Limitations for OSPFv2

OSPFv2 has the following configuration guidelines and limitations:

- If you enter the **no graceful-restart planned only** command, graceful restart is disabled.

- Cisco NX-OS displays areas in dotted decimal notation regardless of whether you enter the area in decimal or dotted decimal notation.
- All OSPFv2 routers must operate in the same RFC compatibility mode. OSPFv2 for Cisco Nexus® 3550-T switch complies with RFC 2328. Use the **rfc1583compatibility** command in router configuration mode if your network includes routers that support only RFC 1583.
- In scaled scenarios, when the number of interfaces and link-state advertisements in an OSPF process is large, the snmp-walk on OSPF MIB objects is expected to time out with a small-values timeout at the SNMP agent. If your observe a timeout on the querying SNMP agent while polling OSPF MIB objects, increase the timeout value on the polling SNMP agent.
- The following guidelines and limitations apply to the administrative distance feature:
  - When an OSPF route has two or more equal cost paths, configuring the administrative distance is non-deterministic for the **match ip route-source** command.
  - Configuring the administrative distance is supported only for the **match route-type**, **match ip address prefix-list**, and **match ip route-source prefix-list** commands. The other match statements are ignored.
  - There is no preference among the **match route-type**, **match ip address**, and **match ip route-source** commands for setting the administrative distance of OSPF routes. In this way, the behavior of the table map for setting the administrative distance in Cisco Nexus® 3550-T switch OSPF is different from that in Cisco IOS OSPF.
  - The discard route is always assigned an administrative distance of 220. No configuration in the table map applies to OSPF discard routes.
- The output of the **show run ospf** command might show the default values for some OSPF commands.
- Cisco Nexus® 3550-T switch does not forward OSPF neighbor discovery packets, OSPF neighbors are not discovered when Cisco Nexus® 3550-T is an intermediate switch.

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

# Default Settings for OSPFv2

The table lists the default settings for OSPFv2 parameters.

***Table 3: Default OSPFv2 Parameters***

| Parameters | Default |
|---|---|
| Administrative distance | 110 |
| Hello interval | 10 seconds |
| Dead interval | 40 seconds |

| Parameters | Default |
|---|---|
| Discard routes | Enabled |
| Graceful restart grace period | 60 seconds |
| OSPFv2 feature | Disabled |
| Stub router advertisement announce time | 600 seconds |
| Reference bandwidth for link cost calculation | 10 Gb/s |
| LSA minimal arrival time | 1000 milliseconds |
| LSA group pacing | 10 seconds |
| SPF calculation initial delay time | 200 milliseconds |
| SPF minimum hold time | 5000 milliseconds |
| SPF calculation initial delay time | 1000 milliseconds |

# Configuring Basic OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

## Enabling OSPFv2

You must enable the OSPFv2 feature before you can configure OSPFv2.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **feature ospf**<br>**Example:**<br>`switch(config)# feature ospf`<br>**Example:** | Enables the OSPFv2 feature. |
| **Step 3** | (Optional) **show feature**<br>**Example:**<br>`switch(config)# show feature` | Displays enabled and disabled features. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

To disable the OSPFv2 feature and remove all associated configuration, use the no feature ospf command in global configuration mode:

| Command | Purpose |
|---|---|
| **no feature ospf**<br>**Example:**<br>`switch(config)# no feature ospf` | Disables the OSPFv2 feature and removes all associated configuration. |

# Creating an OSPFv2 Instance

The first step in configuring OSPFv2 is to create an OSPFv2 instance. You assign a unique instance tag for this OSPFv2 instance. The instance tag can be any string.

For more information about OSPFv2 instance parameters, see the Configuring Advanced OSPFv2, on page 50 section.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

Use the **show ip ospf** *instance-tag* command to verify that the instance tag is not in use.

OSPFv2 must be able to obtain a router identifier (for example, a configured loopback address) or you must configure the router ID option.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | [**no**]**router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | (Optional) **router-id** *ip-address*<br>**Example:** | Configures the OSPFv2 router ID. This IP address identifies this OSPFv2 instance and must exist on a configured interface in the system. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router)# router-id 192.0.2.1` | |
| **Step 4** | (Optional) **show ip ospf** *instance-tag*<br>**Example:**<br>`switch(config-router)# show ip ospf 201` | Displays OSPF information. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

To remove the OSPFv2 instance and all associated configuration, use the no router ospf command in global configuration mode.

| Command | Purpose |
|---|---|
| **no router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# no router ospf 201` | Deletes the OSPF instance and the associated configuration. |

**Note** This command does not remove the OSPF configuration in interface mode. You must manually remove any OSPFv2 commands configured in interface mode.

# Configuring Optional Parameters on an OSPFv2 Instance

You can configure optional parameters for OSPF, see the Configuring Advanced OSPFv2, on page 50 section.

You can configure the following optional parameters for OSPFv2 in router configuration mode:

### Before you begin

Ensure that you have enabled the OSPF feature, (see the Enabling OSPFv2, on page 40 section).

OSPFv2 must be able to obtain a router identifier (for example, a configured loopback address) or you must configure the router ID option.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **distance** *number*<br>**Example:**<br>`switch(config-router)# distance 25` | Configures the administrative distance for this OSPFv2 instance. The range is from 1 to 255. The default is 110. |
| **Step 2** | **log-adjacency-changes [detail]**<br>**Example:**<br>`switch(config-router)# log-adjacency-changes` | Generates a system message whenever a neighbor changes state. |
| **Step 3** | **maximum-paths** *path-number*<br>**Example:**<br>`switch(config-router)# maximum-paths 4` | Configures the maximum number of equal OSPFv2 paths to a destination in the route table. This command is used for load balancing. The range is from 1 to 16. The default is 8. |
| **Step 4** | **distance** *number*<br>**Example:**<br>`switch(config-router)# distance 25` | Configures the administrative distance for this OSPFv2 instance. The range is from 1 to 255. The default is 110. |
| **Step 5** | **log-adjacency-changes** [**detail**]<br>**Example:**<br>`switch(config-router)# log-adjacency-changes` | Generates a system message whenever a neighbor changes state. |
| **Step 6** | **maximum-paths** *path-number*<br>**Example:**<br>`switch(config-router)# maximum-paths 4` | Configures the maximum number of equal OSPFv2 paths to a destination in the route table. This command is used for load balancing. The range is from 1 to 16. The default is 8. |
| **Step 7** | **passive-interface default**<br>**Example:**<br>`switch(config-router)# passive-interface default` | Suppresses routing updates on all interfaces. This command is overridden by the VRF or interface command mode configuration. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to create an OSPFv2 instance:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# copy running-config startup-config
```

# Configuring Networks in OSPFv2

You can configure a network to OSPFv2 by associating it through the interface that the router uses to connect to that network (see the Neighbors section). You can add all networks to the default backbone area (Area 0), or you can create new areas using any decimal number or an IP address.

**Note** All areas must connect to the backbone area either directly or through a virtual link.

**Note** OSPF is not enabled on an interface until you configure a valid IP address for that interface.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/2`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **ip address** *ip-prefix/length*<br>**Example:**<br>`switch(config-if)# ip address 192.0.2.1/16` | Assigns an IP address and subnet mask to this interface. |
| **Step 4** | **ip router ospf** *instance-tag* **area** *area-id* [**secondaries none**]<br>**Example:**<br>`switch(config-if)# ip router ospf 201 area 0.0.0.15` | Adds the interface to the OSPFv2 instance and area. |
| **Step 5** | (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config-if)# show ip ospf 201 interface ethernet 1/2` | Displays OSPF information. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Saves this configuration change. |
| **Step 7** | (Optional) **ip ospf cost** *number*<br>**Example:**<br>`switch(config-if)# ip ospf cost 25` | Configures the OSPFv2 cost metric for this interface. The default is to calculate cost metric, based on reference bandwidth and interface bandwidth. The range is from 1 to 65535. |
| **Step 8** | (Optional) **ip ospf dead-interval** *seconds*<br>**Example:**<br>`switch(config-if)# ip ospf dead-interval 50` | Configures the OSPFv2 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds. |
| **Step 9** | (Optional) **ip ospf hello-interval** *seconds*<br>**Example:**<br>`switch(config-if)# ip ospf hello-interval 25` | Configures the OSPFv2 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds. |
| **Step 10** | (Optional) **[default \| no] ip ospf passive-interface**<br>**Example:**<br>`switch(config-if)# ip ospf passive-interface` | Suppresses routing updates on the interface. This command overrides the router or VRF command mode configuration. The **default** option removes this interface mode command and reverts to the router or VRF configuration, if present. |
| **Step 11** | (Optional) **ip ospf priority** *number*<br>**Example:**<br>`switch(config-if)# ip ospf priority 25` | Configures the OSPFv2 priority, used to determine the DR for an area. The range is from 0 to 255. The default is 1. See the Designated Routers section. |
| **Step 12** | (Optional) **ip ospf shutdown**<br>**Example:**<br>`switch(config-if)# ip ospf shutdown` | Shuts down the OSPFv2 instance on this interface. |

### Example

This example shows how to add a network area 0.0.0.10 in OSPFv2 instance 201:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0.0.0.10
switch(config-if)# copy running-config startup-config
```

Use the **show ip ospf interface** command to verify the interface configuration. Use the **show ip ospf neighbor** command to see the neighbors for this interface.

# Configuring Authentication for an Area

You can configure authentication for all networks in an area or for individual interfaces in the area. Interface authentication configuration overrides area authentication.

### Before you begin

Ensure that you have enabled the OSPF feature , see the Enabling OSPFv2 section.

Ensure that all neighbors on an interface share the same authentication configuration, including the shared authentication key.

Create the key chain for this authentication configuration. See the *Cisco Nexus® 3550-T Security Configuration* section.

**Note** For OSPFv2, the key identifier in the **key** *key-id* command supports values from 2 to 255 only.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **area** *area-id* **authentication** [**message-digest**]<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 authentication` | Configures the authentication mode for an area. |
| **Step 4** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config-router)# interface ethernet 1/2`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 5** | (Optional) **ip ospf authentication-key** [**0** \| **3**] *key*<br>**Example:**<br>`switch(config-if)# ip ospf authentication-key 0 mypass` | Configures simple password authentication for this interface. Use this command if the authentication is not set to key-chain or message-digest. 0 configures the password in clear text. 3 configures the password as 3DES encrypted. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | (Optional) **ip ospf message-digest-key** *key-id* **md5** [**0** \| **3**] *key*<br>**Example:**<br>`switch(config-if)# ip ospf message-digest-key 21 md5 0 mypass` | Configures message digest authentication for this interface. Use this command if the authentication is set to message-digest. The key-id range is from 1 to 255. The MD5 option 0 configures the password in clear text and 3 configures the pass key as 3DES encrypted. |
| **Step 7** | (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config-if)# show ip ospf 201 interface ethernet 1/2` | Displays OSPF information. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`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 Authentication for an Interface

You can configure authentication for all networks in an area or for individual interfaces in the area. Interface authentication configuration overrides area authentication.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

Ensure that all neighbors on an interface share the same authentication configuration, including the shared authentication key.

Create the key chain for this authentication configuration. See the *Cisco Nexus® 3550-T Security Configuration* section.

**Note** For OSPFv2, the key identifier in the **key** *key-id* command supports values from 2 to 255 only.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:** | Enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# interface ethernet 1/2`<br>`switch(config-if)#` | |
| **Step 3** | **ip ospf authentication** [**message-digest**]<br>**Example:**<br>`switch(config-if)# ip ospf authentication` | Enables interface authentication mode for OSPFv2 for either cleartext or message-digest type. Overrides area-based authentication for this interface. All neighbors must share this authentication type. |
| **Step 4** | (Optional) **ip ospf authentication key-chain** *key-id*<br>**Example:**<br>`switch(config-if)# ip ospf authentication key-chain Test1` | Configures interface authentication to use key chains for OSPFv2. See the *Cisco Standalone Series NX-OS Security Configuration Guide*, for details on key chains. |
| **Step 5** | (Optional) **ip ospf authentication-key** [**0** \| **3** \| **7**] *key*<br>**Example:**<br>`switch(config-if)# ip ospf authentication-key 0 mypass` | Configures simple password authentication for this interface. Use this command if the authentication is not set to key-chain or message-digest.<br>The options are as follows:<br>• 0—Configures the password in clear text.<br>• 3—Configures the pass key as 3DES encrypted.<br>• 7—Configures the key as Cisco type 7 encrypted. |
| **Step 6** | (Optional) **ip ospf message-digest-key** *key-id* **md5** [**0** \| **3** \| **7**] *key*<br>**Example:**<br>`switch(config-if)# ip ospf message-digest-key 21 md5 0 mypass` | Configures message digest authentication for this interface. Use this command if the authentication is set to message-digest. The key-id range is from 1 to 255. The MD5 options are as follows:<br>• 0—Configures the password in clear text.<br>• 3—Configures the pass key as 3DES encrypted.<br>• 7—Configures the key as Cisco type 7 encrypted. |
| **Step 7** | (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config-if)# show ip ospf 201 interface ethernet 1/2` | Displays OSPF information. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to set an interface for simple, unencrypted passwords and set the password for Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# ip router ospf 201 area 0.0.0.10
switch(config-if)# ip ospf authentication
switch(config-if)# ip ospf authentication-key 0 mypass
switch(config-if)# copy running-config startup-config
```

This example shows how to configure OSPFv2 HMAC-SHA-1 and MD5 cryptographic authentication:

```
switch# configure terminal
switch(config)# key chain chain1
switch(config-keychain)# key 1
switch(config-keychain-key)# key-string 7 070724404206
switch(config-keychain-key)# accept-lifetime 01:01:01 Jan 01 2015 infinite
switch(config-keychain-key)# send-lifetime 01:01:01 Jan 01 2015 infinite
switch(config-keychain-key)# cryptographic-algorithm HMAC-SHA-1
switch(config-keychain-key)# exit
switch(config-keychain)# key 2
switch(config-keychain-key)# key-string 7 070e234f1f5b4a
switch(config-keychain-key)# accept-lifetime 10:51:01 Jul 24 2015 infinite
switch(config-keychain-key)# send-lifetime 10:51:01 Jul 24 2015 infinite
switch(config-keychain-key)# cryptographic-algorithm MD5
switch(config-keychain-key)# exit
switch(config-keychain)# exit

switch(config)# interface ethernet 1/1
switch(config-if)# ip router ospf 1 area 0.0.0.0
switch(config-if)# ip ospf authentication message-digest
switch(config-if)# ip ospf authentication key-chain chain1

switch(config-if)# show key chain chain1
Key-Chain chain1
Key 1 -- text 7 "070724404206"
cryptographic-algorithm HMAC-SHA-1
accept lifetime UTC (01:01:01 Jan 01 2015)-(always valid) [active]
send lifetime UTC (01:01:01 Jan 01 2015)-(always valid) [active]
Key 2 -- text 7 "070e234f1f5b4a"
cryptographic-algorithm MD
accept lifetime UTC (10:51:00 Jul 24 2015)-(always valid) [active]
send lifetime UTC (10:51:00 Jul 24 2015)-(always valid) [active]

switch(config-if)# show ip ospf interface ethernet 1/1
Ethernet1/1 is up, line protocol is up
IP address 11.11.11.1/24
Process ID 1 VRF default, area 0.0.0.3
```

```
Enabled by interface configuration
State BDR, Network type BROADCAST, cost 40
Index 6, Transmit delay 1 sec, Router Priority 1
Designated Router ID: 33.33.33.33, address: 11.11.11.3
Backup Designated Router ID: 1.1.1.1, address: 11.11.11.1
2 Neighbors, flooding to 2, adjacent with 2
Timer intervals: Hello 10, Dead 40, Wait 40, Retransmit 5
Hello timer due in 00:00:08
Message-digest authentication, using keychain key1 (ready)
Sending SA: Key id 2, Algorithm MD5
Number of opaque link LSAs: 0, checksum sum 0
```

# Configuring Advanced OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

## Configuring Filter Lists for Border Routers

You can separate your OSPFv2 domain into a series of areas that contain related networks. All areas must connect to the backbone area through an area border router (ABR). OSPFv2 domains can connect to external domains as well, through an autonomous system border router (ASBR).

ABRs have the following optional configuration parameters:

- Area range—Configures route summarization between areas. See the Configuring Route Summarization, on page 62 section.
- Filter list—Filters the Network Summary (type 3) LSAs that are allowed in from an external area.

ASBRs also support filter lists.

### Before you begin

Ensure that you have enabled the OSPF feature. See the Enabling OSPFv2, on page 40 section).

Create the route map that the filter list uses to filter IP prefixes in incoming or outgoing Network Summary (type 3) LSAs. See the Configuring Route Policy Manager section, for more information. See the Areas, on page 30 section.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **area** *area-id* **filter-list route-map** *map-name* {**in** \| **out**}<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 filter-list route-map FilterLSAs in` | Filters incoming or outgoing Network Summary (type 3) LSAs on an ABR. |
| **Step 4** | (Optional) **show ip ospf policy statistics area** *id* **filter-list** {**in** \| **out**}<br>**Example:**<br>`switch(config-router)# show ip ospf policy statistics area 0.0.0.10 filter-list in` | Displays OSPF policy information. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to configure a filter list in area 0.0.0.10:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 filter-list route-map FilterLSAs in
switch(config-router)# copy running-config startup-config
```

# Configuring Stub Areas

You can configure a stub area for part of an OSPFv2 domain where external traffic is not necessary. Stub areas block AS External (type 5) LSAs and limit unnecessary routing to and from selected networks. See the Stub Area, on page 34 section. You can optionally block all summary routes from going into the stub area.

### Before you begin

Ensure that you have enabled the OSPF feature. (see the Enabling OSPFv2, on page 40 section).

Ensure that there are no virtual links or ASBRs in the proposed stub area.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **area** *area-id* **stub**<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 stub` | Creates this area as a stub area. |
| **Step 4** | (Optional) **area** *area-id* **default-cost** *cost*<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 default-cost 25` | Sets the cost metric for the default summary route sent into this stub area. The range is from 0 to 16777215. The default is 1. |
| **Step 5** | (Optional) **show ip ospf** *instance-tag*<br>**Example:**<br>`switch(config-router)# show ip ospf 201` | Displays OSPF information. |
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to create a stub area:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 stub
switch(config-router)# copy running-config startup-config
```

# Configuring a Totally Stubby Area

You can create a totally stubby area and prevent all summary route updates from going into the stub area.

To create a totally stubby area, use the following command in router configuration mode:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **area** *area-id* **stub no-summary**<br>**Example:** | Creates this area as a totally stubby area. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router)# area 20 stub no-summary` | |

# Configuring NSSA

You can configure an NSSA for part of an OSPFv2 domain where limited external traffic is required. You can optionally translate this external traffic to an AS External (type 5) LSA and flood the OSPFv2 domain with this routing information. An NSSA can be configured with the following optional parameters:

- No redistribution—Redistributed routes bypass the NSSA and are redistributed to other areas in the OSPFv2 autonomous system. Use this option when the NSSA ASBR is also an ABR.
- Default information originate—Generates an NSSA External (type 7) LSA for a default route to the external autonomous system. Use this option on an NSSA ASBR if the ASBR contains the default route in the routing table. This option can be used on an NSSA ABR whether or not the ABR contains the default route in the routing table.
- Route map—Filters the external routes so that only those routes that you want are flooded throughout the NSSA and other areas.
- No summary—Blocks all summary routes from flooding the NSSA. Use this option on the NSSA ABR.
- Translate—Translates NSSA External LSAs to AS External LSAs for areas outside the NSSA. Use this command on an NSSA ABR to flood the redistributed routes throughout the OSPFv2 autonomous system. You can optionally suppress the forwarding address in these AS External LSAs. If you choose this option, the forwarding address is set to 0.0.0.0.

**Note** The translate option requires a separate **area** *area-id* **nssa**command, preceded by the **area** *area-id* **nssa** command that creates the NSSA and configures the other options.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

Ensure that there are no virtual links in the proposed NSSA and that it is not the backbone area.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:** | Creates a new OSPFv2 instance with the configured instance tag. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# router ospf 201`<br>`switch(config-router)#` | |
| **Step 3** | **area** *area-id* **nssa [no-redistribution] [default-information-originate]originate** [**route-map** *map-name*]] [**no-summary**]<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 nssa no-redistribution` | Creates this area as an NSSA. |
| **Step 4** | (Optional) **area** *area-id* **nssa translate type7** {**always** \| **never**} [**suppress-fa**]<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 nssa translate type7 always` | Configures the NSSA to translate AS External (type 7) LSAs to NSSA External (type 5) LSAs. |
| **Step 5** | (Optional) **area** *area-id* **default-cost** *cost*<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 default-cost 25` | Sets the cost metric for the default summary route sent into this NSSA. |
| **Step 6** | (Optional) **show ip ospf** *instance-tag*<br>**Example:**<br>`switch(config-router)# show ip ospf 201` | Displays OSPF information. |
| **Step 7** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to create an NSSA that blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa no-summary
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that generates a default route:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa default-info-originate
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA that filters external routes and blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospf 201
```

```
switch(config-router)# area 0.0.0.10 nssa route-map ExternalFilter no-summary
switch(config-router)# copy running-config startup-config
```

This example shows how to create an NSSA and then configure the NSSA to always translate AS External (type 7) LSAs to NSSA External (type 5) LSAs:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa
switch(config-router)# area 0.0.0.10 nssa translate type 7 always
switch(config-router)# copy running-config startup-config
```

# Configuring Multi-Area Adjacency

You can add more than one area to an existing OSPFv2 interface. The additional logical interfaces support multi-area adjacency.

### Before you begin

You must enable OSPFv2 (see the Enabling OSPFv2, on page 40 section).

Ensure that you have configured a primary area for the interface (see the Configuring Networks in OSPFv2, on page 44 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/2`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **ip router ospf** [*instance-tag*] **multi-area** *area-id*<br>**Example:**<br>`switch(config-if)# ip router ospf 201 multi-area 3` | Adds the interface to another area.<br>**Note** The *instance-tag* argument is optional. If you do not specify an instance, the multi-area configuration is applied to the same instance that is configured for the primary area on that interface. |
| **Step 4** | (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config-if)# show ip ospf 201 interface ethernet 1/2` | Displays OSPFv2 information. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to add a second area to an OSPFv2 interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0.0.0.10
switch(config-if)# ip router ospf 201 multi-area 20
switch(config-if)# copy running-config startup-config
```

# Configuring Virtual Links

A virtual link connects an isolated area to the backbone area through an intermediate area. See the Virtual Links section.You can configure the following optional parameters for a virtual link:

- Authentication—Sets a simple password or MD5 message digest authentication and associated keys.
- Dead interval—Sets the time that a neighbor waits for a Hello packet before declaring the local router as dead and tearing down adjacencies.
- Hello interval—Sets the time between successive Hello packets.
- Retransmit interval—Sets the estimated time between successive LSAs.
- Transmit delay—Sets the estimated time to transmit an LSA to a neighbor.

**Note** You must configure the virtual link on both routers involved before the link becomes active.

You cannot add a virtual link to a stub area.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **area** *area-id* **virtual link** *router-id*<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 virtual-link 10.1.2.3`<br>`switch(config-router-vlink)#` | Creates one end of a virtual link to a remote router. You must create the virtual link on that remote router to complete the link. |
| **Step 4** | (Optional) **show ip ospf virtual-link** [**brief**]<br>**Example:**<br>`switch(config-router-vlink)# show ip ospf`<br>`virtual-link` | Displays OSPF virtual link information. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |
| **Step 6** | (Optional) **authentication** [**key-chain** *key-id* **message-digest** \| **null**]<br>**Example:**<br>`switch(config-router-vlink)# authentication message-digest` | Overrides area-based authentication for this virtual link. |
| **Step 7** | (Optional) **authentication-key** [**0** \| **3**] *key*<br>**Example:**<br>`switch(config-router-vlink)# authentication-key 0 mypass` | Configures a simple password for this virtual link. Use this command if the authentication is not set to key-chain or message-digest. 0 configures the password in clear text. 3 configures the password as 3DES encrypted. |
| **Step 8** | (Optional) **dead-interval** *seconds*<br>**Example:**<br>`switch(config-router-vlink)# dead-interval 50` | Configures the OSPFv2 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds. |
| **Step 9** | (Optional) **hello-interval** *seconds*<br>**Example:**<br>`switch(config-router-vlink)# hello-interval 25` | Configures the OSPFv2 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds. |
| **Step 10** | (Optional) **message-digest-key** *key-id* **md5** [**0** \| **3**] *key*<br>**Example:** | Configures message digest authentication for this virtual link. Use this command if the authentication is set to message-digest. 0 |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router-vlink)# message-digest-key 21 md5 0 mypass` | configures the password in clear text. 3 configures the pass key as 3DES encrypted. |
| **Step 11** | (Optional) **retransmit-interval** *seconds*<br>**Example:**<br>`switch(config-router-vlink)# retransmit-interval 50` | Configures the OSPFv2 retransmit interval, in seconds. The range is from 1 to 65535. The default is 5. |
| **Step 12** | (Optional) **transmit-delay** *seconds*<br>**Example:**<br>`switch(config-router-vlink)# transmit-delay 2` | Configures the OSPFv2 transmit-delay, in seconds. The range is from 1 to 450. The default is 1. |

### Example

This example shows how to create a simple virtual link between two ABRs.

The configuration for ABR 1 (router ID 27.0.0.55) is as follows:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 virtual-link 10.1.2.3
switch(config-router)# copy running-config startup-config
```

The configuration for ABR 2 (Router ID 10.1.2.3) is as follows:

```
switch# configure terminal
switch(config)# router ospf 101
switch(config-router)# area 0.0.0.10 virtual-link 27.0.0.55
switch(config-router)# copy running-config startup-config
```

# Configuring Redistribution

You can redistribute routes that are learned from other routing protocols into an OSPFv2 autonomous system through the ASBR.

For redistributing the default route, you must specify the following parameter:

- Default information originate—Generates an autonomous system External (type 5) LSA for a default route to the external autonomous system.

**Note** Default information originate ignores **match** statements in the optional route map.

For non-default routes, you can configure the following optional parameters for route redistribution in OSPF:

- Default metric—Sets all redistributed routes to the same cost metric.

**Note** If you redistribute static routes, Cisco NX-OS requires the **default-information originate** command to successfully redistribute the default static route.

### Before you begin

Enable the OSPF feature. See Enabling OSPFv2, on page 40.

Create the necessary route maps used for redistribution.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **redistribute** {**bgp** *id* \| **direct** \| **eigrp** *id* \| **isis** *id* \| **ospf** *id* \| **rip** *id* \| **static**} **route-map** *map-name*<br>**Example:**<br>`switch(config-router)# redistribute bgp route-map FilterExternalBGP` | Redistributes the selected protocol into OSPF through the configured route map.<br>**Note** If you redistribute static routes, Cisco NX-OS also redistributes the default static route. |
| **Step 4** | **default-information originate** [**always**] [**route-map** *map-name*]<br>**Example:**<br>`switch(config-router)# default-information-originate route-map DefaultRouteFilter` | Creates a default route into this OSPF domain if the default route exists in the RIB. Use the following optional keywords:<br>• **always**—Always generate the default route of 0.0.0. even if the route does not exist in the RIB.<br>• **route-map**—Generate the default route if the route map returns true.<br>**Note** This command ignores **match** statements in the route map. |
| **Step 5** | **default-metric** [*cost*]<br>**Example:**<br>`switch(config-router)# default-metric 25` | Sets the cost metric for the redistributed routes. This command does not apply to directly connected routes. Use a route map to set the default metric for directly connected routes. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to redistribute the Border Gateway Protocol (BGP) into OSPF:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# copy running-config startup-config
```

# Limiting the Number of Redistributed Routes

Route redistribution can add many routes to the OSPFv2 route table. You can configure a maximum limit to the number of routes accepted from external protocols. OSPFv2 provides the following options to configure redistributed route limits:

- Fixed limit—Logs a message when OSPFv2 reaches the configured maximum. OSPFv2 does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where OSPFv2 logs a warning when that threshold is passed.
- Warning only—Logs a warning only when OSPFv2 reaches the maximum. OSPFv2 continues to accept redistributed routes.
- Withdraw—Starts the timeout period when OSPFv2 reaches the maximum. After the timeout period, OSPFv2 requests all redistributed routes if the current number of redistributed routes is less than the maximum limit. If the current number of redistributed routes is at the maximum limit, OSPFv2 withdraws all redistributed routes. You must clear this condition before OSPFv2 accepts more redistributed routes.
- You can optionally configure the timeout period.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:** | Creates a new OSPFv2 instance with the configured instance tag. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# router ospf 201`<br>`switch(config-router)#` | |
| **Step 3** | **redistribute** {**bgp** *id* \| **direct** \| **eigrp** *id* \| **isis** *id* \| **ospf** *id* \| **rip** *id* \| **static**} **route-map** *map-name*<br>**Example:**<br>`switch(config-router)# redistribute bgp route-map FilterExternalBGP` | Redistributes the selected protocol into OSPF through the configured route map. |
| **Step 4** | **redistribute maximum-prefix** *max* [*threshold*] [**warning-only \| withdraw** [*num-retries timeout*]]<br>**Example:**<br>`switch(config-router)# redistribute maximum-prefix 1000 75 warning-only` | Specifies a maximum number of prefixes that OSPFv2 distributes. The range is from 0 to 65536. Optionally specifies the following:<br>• *threshold*—Percentage of maximum prefixes that trigger a warning message.<br>• **warning-only**—Logs a warning message when the maximum number of prefixes is exceeded.<br>• **withdraw**—Withdraws all redistributed routes. Optionally tries to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* range is 60 to 600 seconds. The default is 300 seconds. Use the **clear ip ospf redistribution** command if all routes are withdrawn. |
| **Step 5** | (Optional) **show running-config ospf**<br>**Example:**<br>`switch(config-router)# show running-config ospf` | Displays the OSPFv2 configuration. |
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to limit the number of redistributed routes into OSPF:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# redistribute maximum-prefix 1000 75
```

# Configuring Route Summarization

You can configure route summarization for inter-area routes by configuring an address range that is summarized. You can also configure route summarization for external, redistributed routes by configuring a summary address for those routes on an ASBR. For more information, see the Route Summarization, on page 36 section.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **area** *area-id* **range** *ip-prefix/length* [**no-advertise**] [**cost** *cost*]<br>**Example:**<br>`switch(config-router)# area 0.0.0.10 range 10.3.0.0/16` | Creates a summary address on an ABR for a range of addresses and optionally does not advertise this summary address in a Network Summary (type 3) LSA. The *cost* range is from 0 to 16777215. |
| **Step 4** | **summary-address** *ip-prefix/length* [**no-advertise** \| **tag** *tag*]<br>**Example:**<br>`switch(config-router)# summary-address 10.5.0.0/16 tag 2` | Creates a summary address on an ASBR for a range of addresses and optionally assigns a tag for this summary address that can be used for redistribution with route maps. |
| **Step 5** | (Optional) **show ip ospf summary-address**<br>**Example:**<br>`switch(config-router)# show ip ospf summary-address` | Displays information about OSPF summary addresses. |
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to create summary addresses between areas on an ABR:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)#  area 0.0.0.10 range 10.3.0.0/16
switch(config-router)# copy running-config startup-config
```

This example shows how to create summary addresses on an ASBR:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# summary-address 10.5.0.0/16
switch(config-router)# copy running-config startup-config
```

# Configuring Stub Route Advertisements

Use stub route advertisements when you want to limit the OSPFv2 traffic through this router for a short time. For more information, see the OSPFv2 Stub Router Advertisements, on page 37 section.

Stub route advertisements can be configured with the following optional parameters:

- On startup—Sends stub route advertisements for the specified announce time.
- Wait for BGP—Sends stub router advertisements until BGP converges.

**Note** You should not save the running configuration of a router when it is configured for a graceful shutdown because the router continues to advertise a maximum metric after it is reloaded.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **max-metric router-lsa** [**external-lsa** [*max-metric-value*]] [**include-stub**] [**on-startup** {*seconds* \| **wait-for bgp** *tag*}] [**summary-lsa** [*max-metric-value*}] <br>**Example:**<br>`switch(config-router)# max-metric router-lsa` | Configures OSPFv2 stub route advertisements. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to enable the stub router advertisements on startup for the default 600 seconds:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# max-metric router-lsa on-startup
switch(config-router)# copy running-config startup-config
```

# Configuring the Administrative Distance of Routes

You can set the administrative distance of routes added by OSPFv2 into the RIB.

The administrative distance is a rating of the trustworthiness of a routing information source. A higher value indicates a lower trust rating. Typically, a route can be learned through more than one routing protocol. The administrative distance is used to discriminate between routes learned from more than one routing protocol. The route with the lowest administrative distance is installed in the IP routing table.

### Before you begin

Ensure that you have enabled OSPF (see the Enabling OSPFv2, on page 40 section).

See the guidelines and limitations for this feature in the Guidelines and Limitations for OSPFv2, on page 38 section.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | [**no**] **table-map** *map-name*<br>**Example:**<br>`switch(config-router)# table-map foo` | Configures the policy for filtering or modifying OSPFv2 routes before sending them to the RIB. You can enter up to 63 alphanumeric characters for the map name. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **exit**<br>**Example:**<br>`switch(config-router)# exit`<br>`switch(config)#` | Exits router configuration mode. |
| **Step 5** | **route-map** *map-name* [**permit** \| **deny**] [*seq*]<br>**Example:**<br>`switch(config)# route-map foo permit 10`<br>`switch(config-route-map)#` | Creates a route map or enters route-map configuration mode for an existing route map. Use *seq* to order the entries in a route map.<br>**Note** The **permit** option enables you to set the distance. If you use the **deny** option, the default distance is applied. |
| **Step 6** | **match route-type** *route-type*<br>**Example:**<br>`switch(config-route-map)# match route-type external` | Matches against one of the following route types:<br>• external—The external route (BGP, EIGRP, and OSPF type 1 or 2)<br>• inter-area—OSPF inter-area route<br>• internal—The internal route (including the OSPF intra- or inter-area)<br>• intra-area—OSPF intra-area route<br>• nssa-external—The NSSA external route (OSPF type 1 or 2)<br>• type-1—The OSPF external type 1 route<br>• type-2—The OSPF external type 2 route |
| **Step 7** | **match ip route-source prefix-list** *name*<br>**Example:**<br>`switch(config-route-map)# match ip route-source prefix-list p1` | Matches the IPv4 route source address or router ID of a route to one or more IP prefix lists. Use the **ip prefix-list** command to create the prefix list. |
| **Step 8** | **match ip address prefix-list** *name*<br>**Example:**<br>`switch(config-route-map)# match ip address prefix-list p1` | Matches against one or more IPv4 prefix lists. Use the **ip prefix-list** command to create the prefix list. |
| **Step 9** | **set distance** *value*<br>**Example:**<br>`switch(config-route-map)# set distance 150` | Sets the administrative distance of routes for OSPFv2. The range is from 1 to 255. |
| **Step 10** | (Optional) **copy running-config startup-config** | Saves this configuration change. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`switch(config-route-map)# copy running-config startup-config` | |

### Example

This example shows how to configure the OSPFv2 administrative distance for inter-area routes to 150, for external routes to 200, and for all prefixes in prefix list p1 to 190:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# table-map foo
switch(config-router)# exit
switch(config)# route-map foo permit 10
switch(config-route-map)# match route-type inter-area
switch(config-route-map)# set distance 150
switch(config-route-map)# exit
switch(config)# route-map foo permit 20
switch(config-route-map)# match route-type external
switch(config-route-map)# set distance 200
switch(config-route-map)# exit
switch(config)# route-map foo permit 30
switch(config-route-map)# match ip route-source prefix-list p1
switch(config-route-map)# match ip address prefix-list p1
switch(config-route-map)# set distance 190
```

# Modifying the Default Timers

OSPFv2 includes a number of timers that control the behavior of protocol messages and shortest path first (SPF) calculations. OSPFv2 includes the following optional timer parameters:

- LSA arrival time—Sets the minimum interval allowed between LSAs that arrive from a neighbor. LSAs that arrive faster than this time are dropped.
- Pacing LSAs—Sets the interval at which LSAs are collected into a group and refreshed, checksummed, or aged. This timer controls how frequently LSA updates occur and optimizes how many are sent in an LSA update message (see the Flooding and LSA Group Pacing, on page 32 section).
- Throttle LSAs—Sets the rate limits for generating LSAs. This timer controls how frequently LSAs are generated after a topology change occurs.
- Throttle SPF calculation—Controls how frequently the SPF calculation is run.

At the interface level, you can also control the following timers:

- Retransmit interval—Sets the estimated time between successive LSAs
- Transmit delay—Sets the estimated time to transmit an LSA to a neighbor.

See the Configuring Networks in OSPFv2, on page 44 section for information about the hello interval and dead timer.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2, on page 40 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **timers lsa-arrival** *msec*<br>**Example:**<br>`switch(config-router)# timers lsa-arrival 2000` | Sets the LSA arrival time in milliseconds. The range is from 10 to 600000. The default is 1000 milliseconds. |
| **Step 4** | **timers lsa-group-pacing** *seconds*<br>**Example:**<br>`switch(config-router)# timers lsa-group-pacing 1800` | Sets the interval in seconds for grouping LSAs. The range is from 1 to 1800. The default is 240 seconds. |
| **Step 5** | **timers throttle lsa** *start-time hold-interval max-time*<br>**Example:**<br>`switch(config-router)# timers throttle lsa 3000 6000 6000` | Sets the rate limit in milliseconds for generating LSAs with the following timers:<br>• *start-time*—The range is from 50 to 5000 milliseconds. The default value is 50 milliseconds.<br>• *hold-interrval*—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds.<br>• *max-time*—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds |
| **Step 6** | **timers throttle spf** *delay-time hold-time max-wait*<br>**Example:**<br>`switch(config-router)# timers throttle spf 3000 2000 4000` | Sets the SPF best path schedule initial delay time and the minimum hold time in seconds between SPF best path calculations. The range is from 1 to 600000. The default is no delay time and 5000 millisecond hold time. |
| **Step 7** | **interface** *type slot/port*<br>**Example:** | Enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# interface ethernet 1/2`<br>`switch(config-if)` | |
| **Step 8** | **ip ospf hello-interval** *seconds*<br>**Example:**<br>`switch(config-if)# ip ospf hello-interval 30` | Sets the hello interval for this interface. The range is from 1 to 65535. The default is 10. |
| **Step 9** | **ip ospf dead-interval** *seconds*<br>**Example:**<br>`switch(config-if)# ip ospf dead-interval 30` | Sets the dead interval for this interface. The range is from 1 to 65535. |
| **Step 10** | **ip ospf retransmit-interval** *seconds*<br>**Example:**<br>`switch(config-if)# ip ospf retransmit-interval 30` | Sets the estimated time in seconds between LSAs transmitted from this interface. The range is from 1 to 65535. The default is 5. |
| **Step 11** | **ip ospf transmit-delay** *seconds*<br>**Example:**<br>`switch(config-if)# ip ospf transmit-delay 450`<br>`switch(config-if)#` | Sets the estimated time in seconds to transmit an LSA to a neighbor. The range is from 1 to 450. The default is 1. |
| **Step 12** | (Optional) **show ip ospf**<br>**Example:**<br>`switch(config-if)# show ip ospf` | Displays information about OSPF. |
| **Step 13** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to control LSA flooding with the lsa-group-pacing option:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# timers lsa-group-pacing 300
switch(config-router)# copy running-config startup-config
```

# Configuring Graceful Restart

Graceful restart is enabled by default. You can configure the following optional parameters for graceful restart in an OSPFv2 instance:

- Grace period—Configures how long neighbors should wait after a graceful restart has started before tearing down adjacencies.
- Helper mode disabled—Disables helper mode on the local OSPFv2 instance. OSPFv2 does not participate in the graceful restart of a neighbor.
- Planned graceful restart only—Configures OSPFv2 to support graceful restart only in the event of a planned restart.

### Before you begin

Ensure that you have enabled OSPF (see the Enabling OSPFv2, on page 40 section).

Ensure that all neighbors are configured for graceful restart with matching optional parameters set.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 3** | **graceful-restart**<br>**Example:**<br>`switch(config-router)# graceful-restart` | Enables a graceful restart. A graceful restart is enabled by default. |
| **Step 4** | (Optional) **graceful-restart grace-period** *seconds*<br>**Example:**<br>`switch(config-router)# graceful-restart grace-period 120` | Sets the grace period, in seconds. The range is from 5 to 1800. The default is 60 seconds. |
| **Step 5** | (Optional) **graceful-restart helper-disable**<br>**Example:**<br>`switch(config-router)# graceful-restart helper-disable` | Disables helper mode. This feature is enabled by default. |
| **Step 6** | (Optional) **graceful-restart planned-only**<br>**Example:**<br>`switch(config-router)# graceful-restart planned-only` | Configures a graceful restart for planned restarts only. |
| **Step 7** | (Optional) **show ip ospf** *instance-tag*<br>**Example:** | Displays OSPF information. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router)# show ip ospf 201` | |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to enable a graceful restart if it has been disabled and set the grace period to 120 seconds:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# graceful-restart
switch(config-router)# graceful-restart grace-period 120
switch(config-router)# copy running-config startup-config
```

# Restarting an OSPFv2 Instance

You can restart an OSPv2 instance. This action clears all neighbors for the instance.

To restart an OSPFv2 instance and remove all associated neighbors, use the following command:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **restart ospf** *instance-tag*<br>**Example:**<br>`switch(config)# restart ospf 201` | Restarts the OSPFv2 instance and removes all neighbors. |

# Configuring OSPFv2 with Virtualization

You can create multiple OSPFv2 instances. You can also create multiple VRFs and use the same or multiple OSPFv2 instances in each VRF. You can assign an OSPFv2 interface to a VRF.

**Note** Configure all other parameters for an interface after you configure the VRF for an interface. Configuring a VRF for an interface deletes all the configuration for that interface.

### Before you begin

Ensure that you have enabled the OSPF feature (see the Enabling OSPFv2 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **vrf context** *vrf-name*<br>**Example:**<br>`switch(config)# vrf context RemoteOfficeVRF`<br>`switch(config-vrf)#` | Creates a new VRF and enters VRF configuration mode. |
| **Step 3** | **router ospf** *instance-tag*<br>**Example:**<br>`switch(config-vrf)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSPFv2 instance with the configured instance tag. |
| **Step 4** | **vrf** *vrf-name*<br>**Example:**<br>`switch(config-router)# vrf RemoteOfficeVRF`<br>`switch(config-router-vrf)#` | Enters VRF configuration mode. |
| **Step 5** | (Optional) **maximum-paths** *path*<br>**Example:**<br>`switch(config-router-vrf)# maximum-paths 4` | Configures the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF. This feature is used for load balancing. |
| **Step 6** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config-router-vrf)# interface ethernet 1/2`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 7** | **vrf member** *vrf-name*<br>**Example:**<br>`switch(config-if)# vrf member RemoteOfficeVRF` | Adds this interface to a VRF. |
| **Step 8** | **ip address** *ip-prefix/length*<br>**Example:**<br>`switch(config-if)# ip address 192.0.2.1/16` | Configures an IP address for this interface. You must do this step after you assign this interface to a VRF. |
| **Step 9** | **ip router ospf** *instance-tag* **area** *area-id*<br>**Example:** | Assigns this interface to the OSPFv2 instance and area configured. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-if)# ip router ospf 201 area 0` | |
| **Step 10** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves the change persistently through reboots and restarts by copying the running configuration to the startup configuration. |

### Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config)# router ospf 201
switch(config)# interface ethernet 1/2
switch(config-if)# vrf member NewVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0
switch(config-if)# copy running-config startup-config
```

# Verifying the OSPFv2 Configuration

To display the OSPFv2 configuration, perform one of the following tasks:

| Command | Purpose |
|---|---|
| **show ip ospf** [*instance-tag*] [**vrf** *vrf-name*] | Displays information about one or more OSPF routing instances. The output includes the following area-level counts:<br>• Interfaces in this area—A count of all interfaces added to this area (configured interfaces).<br>• Active interfaces—A count of all interfaces considered to be in router link states and SPF (UP interfaces).<br>• Passive interfaces—A count of all interfaces considered to be OSPF passive (no adjacencies will be formed).<br>• Loopback interfaces—A count of all local loopback interfaces. |
| **show ip ospf border-routers** [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays the OSPFv2 border router configuration. |
| **show ip ospf database** [ **vrf** { *vrf-name* \| **all** \| **default** \| **management**}] | Displays the OSPFv2 link-state database summary. |

| Command | Purpose |
|---|---|
| **show ip ospf interface** *number* [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays OSPFv2-related interface information. |
| **show ip ospf lsa-content-changed-list** *neighbor-id interface - type number* [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays the OSPFv2 LSAs that have changed. |
| **show ip ospf neighbors** [ *neighbor-id* ] [ **detail** ] [ *interface - type number* ] [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] [ **summary** ] | Displays the list of OSPFv2 neighbors. |
| **show ip ospf request-list** *neighbor-id interface - type number* [ **vrf** {*vrf-name* \| **all** \| **default** \| **management** }] | Displays the list of OSPFv2 link-state requests. |
| **show ip ospf retransmission-list** *neighbor-id interface - type number* [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays the list of OSPFv2 link-state retransmissions. |
| **show ip ospf route** [ *ospf-route* ] [ **summary** ] [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays the internal OSPFv2 routes. |
| **show ip ospf summary-address** [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays information about the OSPFv2 summary addresses. |
| **show ip ospf virtual-links** [ **brief** ] [ **vrf** { *vrf-name* \| **all** \| **default** \| **management** }] | Displays information about OSPFv2 virtual links. |
| **show ip ospf vrf** { *vrf-name* \| **all** \| **default** \| **management** } | Displays information about the VRF-based OSPFv2 configuration. |
| **show running-configuration ospf** | Displays the current running OSPFv2 configuration. |

# Monitoring OSPFv2

To display OSPFv2 statistics, use the following commands:

| Command | Purpose |
|---|---|
| **show ip ospf policy statistics area** *area-id* **filter list** {**in** \| **out**} [**vrf** {*vrf-name* \| **all** \| **default** \| **management**}] | Displays the OSPFv2 route policy statistics for an area. |
| **show ip policy statistics redestribute** {**bgp** *id* \| **direct** \| **eigrp** *id* \| **isis** *id* \| **ospf** *id* \| **rip** *id* \| **static**} [**vrf** {*vrf-name* \| **all** \| **default** \| **management**}] | Displays the OSPFv2 route policy statistics. |
| **show ip ospf statistics** [**vrf** {*vrf-name* \| **all** \| **default** \| **management**}] | Displays the OSPFv2 event counters. |
| **show ip ospf traffic** [*interface-type number*] [**vrf** {*vrf-name* \| **all** \| **default** \| **management**}] | Displays the OSPFv2 packet counters. |

# Configuration Examples for OSPFv2

The following example shows how to configure OSPFv2:

```
feature ospf
router ospf 201
 router-id 290.0.2.1
interface ethernet 1/2
 ip router ospf 201 area 0.0.0.10
 ip ospf authentication
 ip ospf authentication-key 0 mypass
```

## OSPF RFC Compatibility Mode Example

The following example shows how to configure OSPF to be compatible with routers that comply with RFC 1583:

**Note** You must configure RFC 1583 compatibility on any VRF that connects to routers running only RFC 1583 compatible OSPF.

```
switch# configure terminal
switch(config)# feature ospf
switch(config)# router ospf Test1
switch(config-router)# rfc1583compatibility
switch(config-router)# vrf A
switch(config-router-vrf)# rfc1583compatibility
```

# Additional References

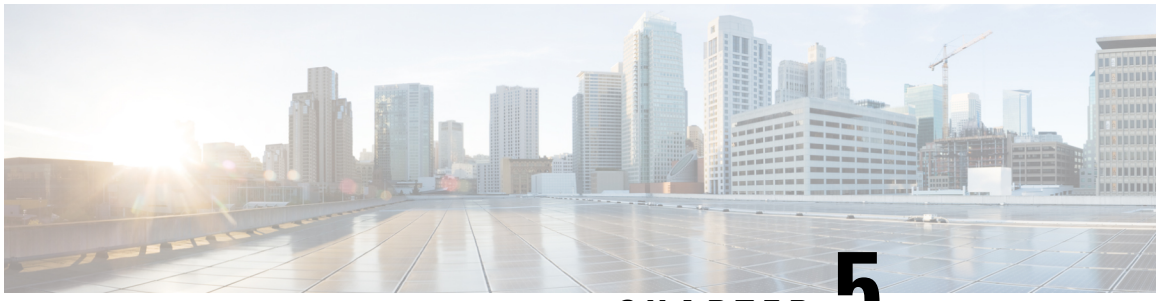
For additional information related to implementing OSPF, see the following sections:

## Related Documents for OSPFv2

| Related Topic | Document Title |
|---|---|
| Keychains | *Cisco Nexus® 3550-T Security Configuration* section |
| Route maps | The Configuring Route Policy Manager section |

## MIBs

| MIBs | MIBs Link |
|---|---|
| MIBs related to OSPFv2 | To locate and download supported MIBs, go to the following URL:<br>3550-T MIBs |



CHAPTER 5

# Configuring Basic BGP

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

This chapter includes the following sections:



## About Basic BGP

Cisco NX-OS supports BGP version 4, which 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.

BGP uses a path-vector routing algorithm to exchange routing information between BGP-enabled networking devices or BGP speakers. Based on this information, each BGP speaker determines a path to reach a particular destination while detecting and avoiding paths with routing loops. The routing information includes the actual route prefix for a destination, the path of autonomous systems to the destination, and other path attributes.

BGP selects a single path, by default, as the best path to a destination host or network. Each path carries well-known mandatory, well-known discretionary, and optional transitive attributes that are used in BGP best-path analysis. You can influence BGP path selection by altering some of these attributes by configuring BGP policies. See the Route Policies and Resetting BGP Sessions, on page 95 section for more information.

# BGP Autonomous Systems

An autonomous system (AS) is a network controlled by a single administration entity. An autonomous system forms a routing domain with one or more interior gateway protocols (IGPs) and a consistent set of routing policies. BGP supports 16-bit and 32-bit autonomous system numbers.

Separate BGP autonomous systems dynamically exchange routing information through external BGP (eBGP) peering sessions. BGP speakers within the same autonomous system can exchange routing information through internal BGP (iBGP) peering sessions.

## 4-Byte AS Number Support

BGP supports 2-byte autonomous system (AS) numbers in plain-text notation or as.dot notation and 4-byte AS numbers in plain-text notation.

When BGP is configured with a 4-byte AS number, the **route-target auto** VXLAN command cannot be used because the AS number along with the VNI (which is already a 3-byte value) is used to generate the route target.

# Administrative Distance

An administrative distance is a rating of the trustworthiness of a routing information source. By default, BGP uses the administrative distances shown in the table.

***Table 4: BGP Default Administrative Distances***

| Distance | Default Value | Function |
|---|---|---|
| External | 20 | Applied to routes learned from eBGP. |
| Internal | 200 | Applied to routes learned from iBGP. |
| Local | 220 | Applied to routes originated by the router. |

**Note** The administrative distance does not influence the BGP path selection algorithm, but it does influence whether BGP-learned routes are installed in the IP routing table.

# BGP Peers

A BGP speaker does not discover another BGP speaker automatically. You must configure the relationships between BGP speakers. A BGP peer is a BGP speaker that has an active TCP connection to another BGP speaker.

## BGP Sessions

BGP uses TCP port 179 to create a TCP session with a peer. When a TCP connection is established between peers, each BGP peer initially exchanges all of its routes—the complete BGP routing table—with the other peer. After this initial exchange, the BGP peers send only incremental updates when a topology change occurs in the network or when a routing policy change occurs. In the periods of inactivity between these updates,

peers exchange special messages called keepalives. The hold time is the maximum time limit that can elapse between receiving consecutive BGP update or keepalive messages.

Cisco NX-OS supports the following peer configuration options:

- Individual IPv4 address—BGP establishes a session with the BGP speaker that matches the remote address and AS number.
- IPv4 prefix peers for a single AS number—BGP establishes sessions with BGP speakers that match the prefix and the AS number.
- Dynamic AS number prefix peers—BGP establishes sessions with BGP speakers that match the prefix and an AS number from a list of configured AS numbers.

### Dynamic AS Numbers for Prefix Peers and Interface Peers

Cisco NX-OS accepts a range or list of AS numbers to establish BGP sessions. For example, if you configure BGP to use IPv4 prefix 192.0.2.0/8 and AS numbers 33, 66, and 99, BGP establishes a session with 192.0.2.1 with AS number 66 but rejects a session from 192.0.2.2 with AS number 50.

Cisco NX-OS does not associate prefix peers with dynamic AS numbers as either interior BGP (iBGP) or external BGP (eBGP) sessions until after the session is established. See the "Configuring Advanced BGP" chapter for more information on iBGP and eBGP.

**Note** The dynamic AS number prefix peer configuration overrides the individual AS number configuration that is inherited from a BGP template. For more information, see the "Configuring Advanced BGP" chapter.

## BGP Router Identifier

To establish BGP sessions between peers, BGP must have a router ID, which is sent to BGP peers in the OPEN message when a BGP session is established. The BGP router ID is a 32-bit value that is often represented by an IPv4 address. You can configure the router ID. By default, Cisco NX-OS sets the router ID to the IPv4 address of a loopback interface on the router. If no loopback interface is configured on the router, the software chooses the highest IPv4 address configured to a physical interface on the router to represent the BGP router ID. The BGP router ID must be unique to the BGP peers in a network.

If BGP does not have a router ID, it cannot establish any peering sessions with BGP peers.

## BGP and the Unicast RIB

BGP communicates with the unicast routing information base (unicast RIB) to store IPv4 routes in the unicast routing table. After selecting the best path, if BGP determines that the best path change needs to be reflected in the routing table, it sends a route update to the unicast RIB.

BGP receives route notifications regarding changes to its routes in the unicast RIB. It also receives route notifications about other protocol routes to support redistribution.

BGP also receives notifications from the unicast RIB regarding next-hop changes. BGP uses these notifications to keep track of the reachability and IGP metric to the next-hop addresses.

Whenever the next-hop reachability or IGP metrics in the unicast RIB change, BGP triggers a best-path recalculation for affected routes.

## BGP Virtualization

BGP supports virtual routing and forwarding (VRF) instances.

# Prerequisites for BGP

BGP has the following prerequisites:

- You must enable BGP (see the Enabling BGP, on page 81 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 configure at least one IGP that is capable of recursive next-hop resolution.
- You must configure an address family under a neighbor for the BGP session establishment.

# Guidelines and Limitations for Basic BGP

BGP has the following configuration guidelines and limitations:

- With sufficient scale (such as - hundreds of peers and thousands of routes per peer) the Graceful Restart mechanism may fail because the default 5 minute stale-path timer might not be enough for BGP convergence to complete before the timer expires. Use the following command to verify the actual time taken for the convergence process:

```
switch# show bgp vrf all all neighbors | in First|RIB
  Last End-of-RIB received 0.022810 after session start
  Last End-of-RIB sent 00:08:36 after session start
  First convergence 00:08:36 after session start with 398002 routes sent
```

- 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 BGP/eBGP multihop sessions.
- Specify a BGP policy if you configure redistribution.

- Define the BGP router ID within a VRF.
- If you decrease the keepalive and hold timer values, you might experience BGP session flaps.
- Although the **show ip bgp** commands are available for verifying the BGP configuration, Cisco recommends that you use the **show bgp** commands instead.
- BGP prefix independent convergence (PIC) edge feature is not supported in Cisco Nexus 3550-T.

# Default Settings

*Table 5: Default BGP Parameters*

| Parameters | Default |
|---|---|
| BGP feature | Disabled |
| Keep alive interval | 60 seconds |
| Hold timer | 180 seconds |
| Auto-summary | Always disabled |
| Synchronization | Always disabled |

# CLI Configuration Modes

The following sections describe how to enter each of the CLI configuration modes for BGP. From a mode, you can enter the **?** command to display the commands available in that mode.

## Global Configuration Mode

Use global configuration mode to create a BGP process and configure advanced features such as AS confederation and route dampening.

This example shows how to enter router configuration mode:

```
switch# configuration
switch(config)# router bgp 64496
switch(config-router)#
```

BGP supports VRF. You can configure BGP within the appropriate VRF if you are using VRFs in your network. See the Configuring Virtualization section for more information.

This example shows how to enter VRF configuration mode:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)#
```

# Neighbor Configuration Mode

Cisco NX-OS provides the neighbor configuration mode to configure BGP peers. You can use neighbor configuration mode to configure all parameters for a peer.

The following example shows how to enter neighbor configuration mode:

```
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.1
switch(config-router-neighbor)#
```

The following example shows how to enter VRF neighbor configuration mode:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# neighbor 192.0.2.1
switch(config-router-vrf-neighbor)#
```

# Neighbor Address Family Configuration Mode

An address family configuration submode inside the neighbor configuration submode is available for entering address family-specific neighbor configuration and enabling the address family for the neighbor. Use this mode for advanced features such as limiting the number of prefixes allowed for this neighbor and removing private AS numbers for eBGP.

This example shows how to enter the IPv4 neighbor address family configuration mode for a neighbor with an IPv4 address:

```
switch(config)# router bgp 64496
switch(config-router# neighbor 192.0.2.1
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)#
```

This example shows how to enter the VRF IPv4 neighbor address family configuration mode or a neighbor with an IPv4 address:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# neighbor 209.165.201.1
switch(config-router-vrf-neighbor)# address-family ipv4 unicast
switch(config-router-vrf-neighbor-af)#
```

# Configuring Basic BGP

To configure a basic BGP, you must enable BGP and configure a BGP peer. Configuring a basic BGP network consists of a few required tasks and many optional tasks. You must configure a BGP routing process and BGP peers.

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

# Enabling BGP

You must enable BGP before you can configure BGP.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters configuration mode. |
| **Step 2** | **[no] feature bgp**<br>**Example:**<br>`switch(config)# feature bgp` | Enables BGP.<br>Use the **no** form of this command to disable this feature. |
| **Step 3** | (Optional) **show feature**<br>**Example:**<br>`switch(config)# show feature` | Displays enabled and disabled features. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves this configuration change. |

# Creating a BGP Instance

You can create a BGP instance and assign a router ID to the BGP instance. For more information, see the BGP Router Identifier, on page 77 section.

### Before you begin

- You must enable BGP (see the Enabling BGP, on page 81 section).
- BGP must be able to obtain a router ID (for example, a configured loopback address).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters configuration mode. |
| **Step 2** | [**no**] **router bgp** *autonomous-system-number*<br>**Example:** | Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# router bgp 64496`<br>`switch(config-router)#` | a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.<br>Use the **no** option with this command to remove the BGP process and the associated configuration. |
| **Step 3** | (Optional) **router-id** *ip-address*<br>**Example:**<br>`switch(config-router)# router-id 192.0.2.255` | Configures the BGP router ID. This IP address identifies this BGP speaker. |
| **Step 4** | (Optional) **address-family** {**ipv4**} {**unicast**\|**multicast**}<br>**Example:**<br>`switch(config-router)# address-family ipv4 unicast`<br>`switch(config-router-af)#` | Enters global address family configuration mode for the IPv4 address family. |
| **Step 5** | (Optional) **network** {*ip-address/length* \| *ip-address* **mask** *mask*} [**route-map** *map-name*]<br>**Example:**<br>`switch(config-router-af)# network 10.10.10.0/24`<br>**Example:**<br>`switch(config-router-af)# network 10.10.10.0 mask 255.255.255.0` | Specifies a network as local to this autonomous system and adds it to the BGP routing table.<br>For exterior protocols, the network command controls which networks are advertised. Interior protocols use the **network** command to determine where to send updates. |
| **Step 6** | (Optional) **show bgp all**<br>**Example:**<br>`switch(config-router-af)# show bgp all` | Displays information about all BGP address families. |
| **Step 7** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-af)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to enable BGP with the IPv4 unicast address family and manually add one network to advertise:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# network 192.0.2.0
switch(config-router-af)# copy running-config startup-config
```

# Restarting a BGP Instance

You can restart a BGP instance and clear all peer sessions for the instance.

To restart a BGP instance and remove all associated peers, use the following command:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **restart bgp***instance-tag*<br>**Example:**<br>`switch(config)# restart bgp 201` | Restarts the BGP instance and resets or reestablishes all peering sessions. |

# Shutting Down BGP

You can shut down the BGP protocol and gracefully disable BGP while retaining the configuration.

To shut down BGP, use the following command in router configuration mode:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **shutdown**<br>**Example:**<br>`switch(config-router)# shutdown` | Restarts the BGP instance and resets or reestablishes all peering sessions. |

# Configuring BGP Peers

You can configure a BGP peer within a BGP process. Each BGP peer has an associated keepalive timer and hold timers. You can set these timers either globally or for each BGP peer. A peer configuration overrides a global configuration.

**Note** You must configure the address family under neighbor configuration mode for each peer.

**Before you begin**

- You must enable BGP (see the Enabling BGP, on page 81 section).

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:** | Enters configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch# configure terminal`<br>`switch(config)#` | |
| **Step 2** | **router bgp** *autonomous-system-number*<br>**Example:**<br>`switch(config)# router bgp 64496`<br>`switch(config-router)#` | Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. |
| **Step 3** | **neighbor** {*ip-address*} **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`<br>`switch(config-router-neighbor)#` | Configures the IPv4 address and AS number for a remote BGP peer. *The ip-address* format is x.x.x.x. The format is A:B::C:D. |
| **Step 4** | **neighbor-as** *as-number*<br>**Example:**<br>`switch(config-router-neighbor)# remote-as 64497` | Configures the AS number for a remote BGP peer. |
| **Step 5** | (Optional) **description***text*<br>**Example:**<br>`switch(config-router-neighbor)# description Peer Router B`<br>`switch(config-router-neighbor)#` | Adds a description for the neighbor. The description is an alphanumeric string up to 80 characters. |
| **Step 6** | (Optional) **timers***keepalive-time hold-time*<br>**Example:**<br>`switch(config-router-neighbor)# timers 30 90` | Adds the keepalive and hold time BGP timer values for the neighbor. The range is from 0 to 3600 seconds. The default is 60 seconds for the keepalive time and 180 seconds for the hold time. |
| **Step 7** | (Optional) **shutdown**<br>**Example:**<br>`switch(config-router-neighbor)# shutdown` | Administratively shuts down this BGP neighbor. This command triggers an automatic notification and session reset for the BGP neighbor sessions. |
| **Step 8** | **address-family**{**ipv4**} {**unicast**\|**multicast**}<br>**Example:**<br>`switch(config-router-neighbor)# address-family ipv4 unicast`<br>`switch(config-router-neighbor-af)#` | Enters neighbor address family configuration mode for the unicast IPv4 address family. |
| **Step 9** | (Optional) **weight** *value*<br>**Example:**<br>`switch(config-router-neighbor-af)# weight 100` | Sets the default weight for routes from this neighbor. The range is from 0 to 65535.<br>All routes learned from this neighbor have the assigned weight initially. The route with the highest weight is chosen as the preferred route when multiple routes are available to a particular network. The weights assigned with |

| | Command or Action | Purpose |
|---|---|---|
| | | the **set weight route-map** command override the weights assigned with this command.<br>If you specify a BGP peer policy template, all the members of the template inherit the characteristics configured with this command. |
| **Step 10** | (Optional) **show bgp** {**ipv4**} {**unicast**\|**multicast**} **neighbors**<br>**Example:**<br>`switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors` | Displays information about BGP peers. |
| **Step 11** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor-af)# copy running-config startup-config` | Saves this configuration change. |

### Example

The following example shows how to configure a BGP peer:

```
switch# configure terminal
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.1 remote-as 64497
switch(config-router-neighbor)# description Peer Router B
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor)# weight 100
switch(config-router-neighbor-af)# copy running-config startup-config
```

# Configuring Dynamic AS Numbers for Prefix Peers

You can configure multiple BGP peers within a BGP process. You can limit BGP session establishment to a single AS number or multiple AS numbers in a route map.

BGP sessions configured through dynamic AS numbers for prefix peers ignore the **ebgp-multihop** command and the **disable-connected-check** command.

You can change the list of AS numbers in the route map, but you must use the no neighbor command to change the route-map name. Changes to the AS numbers in the configured route map affect only new sessions.

### Before you begin

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters configuration mode. |
| **Step 2** | **router bgp** *autonomous-system-number*<br>**Example:**<br>`switch(config)# router bgp 64496`<br>`switch(config-router)#` | Enables BGP and assigns the AS number to the local BGP speaker. The AS number can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. |
| **Step 3** | **neighbor** *prefix* **remote-as route-map** *map-name*<br>**Example:**<br>`switch(config-router)# neighbor 192.0.2.0/8 remote-as routemap BGPPeers`<br>`switch(config-router-neighbor)#` | Configures the IPv4 prefix and a route map for the list of accepted AS numbers for the remote BGP peers. The *prefix* format for IPv4 is x.x.x.x/length. The length range is from 1 to 32.<br>The *map-name* can be any case-sensitive, alphanumeric string up to 63 characters. |
| **Step 4** | **neighbor-as** *as-number*<br>**Example:**<br>`switch(config-router-neighbor)# remote-as 64497` | Configures the AS number for a remote BGP peer. |
| **Step 5** | (Optional) **show bgp** {**ipv4** {**unicast** \| **multicast**} **neighbors**<br>**Example:**<br>`switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors` | Displays information about BGP peers. |
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor-af)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to configure dynamic AS numbers for a prefix peer:

```
switch# configure terminal
switch(config)# route-map BGPPeers
switch(config-route-map)# match as-number 64496, 64501-64510
switch(config-route-map)# match as-number as-path-list List1, List2
switch(config-route-map)# exit
switch(config)# router bgp 64496
switch(config-router)# neighbor 192.0.2.0/8 remote-as route-map BGPPeers
switch(config-router-neighbor)# description Peer Router B
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# copy running-config startup-config
```

# Clearing BGP Information

To clear BGP information, use the following commands:

| Command | Purpose |
|---|---|
| **clear bgp all** {*neighbor* \| * \| *as-number* \| **peer-template** *name* \| *prefix*} [**vrf** *vrf-name*] | Clears one or more neighbors from all address families. * clears all neighbors in all address families. The arguments are as follows:<br>• *neighbor*—IPv4 address of a neighbor.<br>• *as-number*— Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.<br>• *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters.<br>• *prefix*—IPv4 prefix. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear bgp all dampening** [**vrf** *vrf-name*] | Clears route flap dampening networks in all address families. The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear bgp all flap-statistics** [**vrf** *vrf-name*] | Clears route flap statistics in all address families. The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear bgp** {**ipv4** } {**unicast** \| **multicast**} **dampening** [**vrf** *vrf-name*] | Clears route flap dampening networks in the selected address family. The*vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear bgp** {**ipv4** } {**unicast** \| **multicast**} **flap-statistics** [**vrf** *vrf-name*] | Clears route flap statistics in the selected address family. The *vrf-name* can be any case-sensitive, alphanumeric string up to 64 characters. |

| Command | Purpose |
|---|---|
| **clear bgp** {**ipv4** } {*neighbor* \|* \| *as-number* \| **peer-template** *name* \| *prefix*} [**vrf** *vrf-name*] | Clears one or more neighbors from the selected address family. * clears all neighbors in the address family. The arguments are as follows:<br>• *neighbor*—IPv4 address of a neighbor.<br>• *as-number*— Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.<br>• *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters.<br>• *prefix*—IPv4 prefix. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear bgp** {**ip** {**unicast** \| **multicast**}} {*neighbor* \|* \|*as-number* \| **peer-template** *name* \| *prefix*} [**vrf** *vrf-name*] | Clears one or more neighbors. * clears all neighbors in the address family. The arguments are as follows:<br>• *neighbor*—IPv4 address of a neighbor.<br>• *as-number*— Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.<br>• *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters.<br>• *prefix*—IPv4 prefix. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |

| Command | Purpose |
|---|---|
| **clear bgp dampening** [*ip-neighbor* \| *ip-prefix*] [**vrf** *vrf-name*] | Clears route flap dampening in one or more networks. The arguments are as follows:<br>• *ip-neighbor*—IPv4 address of a neighbor.<br>• *ip-prefix*—IPv4. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear bgp flap-statistics** [*ip-neighbor* \| *ip-prefix*] [**vrf** *vrf-name*] | Clears route flap statistics in one or more networks. The arguments are as follows:<br>• *ip-neighbor*—IPv4 address of a neighbor.<br>• *ip-prefix*—IPv4. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear ip mbgp** {**ip** {**unicast** \| **multicast**}} {*neighbor* \| * \| *as-number* \| **peer-template** *name* \| *prefix*} [**vrf** *vrf-name*] | Clears one or more neighbors. * clears all neighbors in the address family. The arguments are as follows:<br>• *neighbor*—IPv4 address of a neighbor.<br>• *as-number*— Autonomous system number. The AS number can be a 16-bit integer or a 32-bit integer in the form of higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format.<br>• *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters.<br>• *prefix*—IPv4 prefix. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |

| Command | Purpose |
|---|---|
| **clear ip mbgp dampening** [*ip-neighbor* \| *ip-prefix*] [**vrf** *vrf-name*] | Clears route flap dampening in one or more networks. The arguments are as follows:<br>• *ip-neighbor*—IPv4 address of a neighbor.<br>• *ip-prefix*—IPv4. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |
| **clear ip mbgp flap-statistics** [*ip-neighbor* \| *ip-prefix*] [**vrf** *vrf-name*] | Clears route flap statistics in one or more networks. The arguments are as follows:<br>• *ip-neighbor*—IPv4 address of a neighbor.<br>• *ip-prefix*—IPv4. All neighbors within that prefix are cleared.<br>• *vrf-name*—VRF name. All neighbors in that VRF are cleared. The name can be any case-sensitive, alphanumeric string up to 64 characters. |

# Verifying the Basic BGP Configuration

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

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

| Command | Purpose |
|---|---|
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address* \| {**dampening dampened-paths** [**regexp** *expression*]} [**vrf** *vrf-name*] | Displays the information for BGP route dampening. Use the **clear bgp dampening** command to clear the route flap dampening information. |
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address* \| **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*] | Displays the BGP route history paths. |
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address* \| **filter-list** *list-name* [**vrf** *vrf-name*] | Displays the information for the BGP filter list. |
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address*] **neighbors** [*ip-address* ] [**vrf** *vrf-name*] | Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors. |
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address* ] **neighbors** [*ip-address* ] {**nexthop** \| **nexthop-database**} [**vrf** *vrf-name*] | Displays the information for the BGP route next hop. |
| **show bgp paths** | Displays the BGP path information. |
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address* ] **policy** *name* [**vrf** *vrf-name*] | Displays the BGP policy information. Use the **clear bgp polic**y command to clear the policy information. |
| **show bgp** {**ipv4** } {**unicast** \| **multicast**} [*ip-address*] **prefix-list** *list-name* [**vrf** *vrf-name*] | Displays the BGP routes that match the prefix list. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **received-paths** [**vrf** *vrf-name*] | Displays the BGP paths stored for soft reconfiguration. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **regexp** *expression* [**vrf** *vrf-name*] | Displays the BGP routes that match the AS_path regular expression. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **route-map** *map-name* [**vrf** *vrf-name*] | Displays the BGP routes that match the route map. |
| **show bgp peer-policy** *name* [**vrf** *vrf-name*] | Displays the information about BGP peer policies. |
| **show bgp peer-session** *name* [**vrf** *vrf-name*]<br>show bgp peer-session | Displays the information about BGP peer sessions. |
| **show bgp peer-template** *name* [**vrf** *vrf-name*] | Displays the information about BGP peer templates. Use the **clear bgp peer-template** command to clear all neighbors in a peer template. |
| **show bgp process** | Displays the BGP process information. |
| **show** {**ipv**} **bgp** [*options*] | Displays the BGP status and configuration information. |
| **show** {**ipv**} **mbgp** [*options*] | Displays the BGP status and configuration information. |
| **show running-configuration bgp** | Displays the current running BGP configuration. |

# Monitoring BGP Statistics

To display BGP statistics, use the following commands:

| Command | Purpose |
|---|---|
| **show bgp** {**ipv4** } {**unicast**} [*ip-address*] **flap-statistics** [**vrf** *vrf-name*] | Displays the BGP route flap statistics. Use the **clear bgp flap-statistics command** to clear these statistics. |
| **show bgp sessions** [**vrf** *vrf-name*] | Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics. |
| **show bgp statistics** | Displays the BGP statistics. |

# Configuration Examples for Basic BGP

This example shows a basic BGP configuration:

```
switch(config)# feature bgp
switch(config)# router bgp 64496
switch(config-router)# neighbor 2001:ODB8:0:1::55 remote-as 64496
switch(config-router-af)# next-hop-self
```

# Related Topics

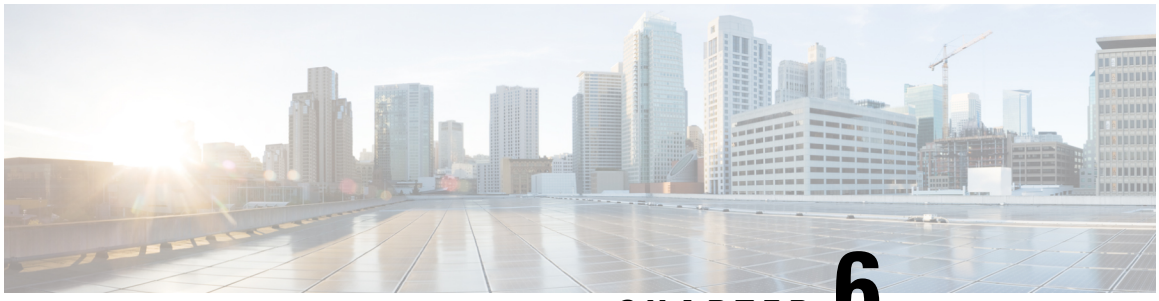
The following topics relate to BGP:

- Configuring Advanced BGP, on page 93
- The *Configuring Route Policy Manager* section

# Where to Go Next

See Configuring Advanced BGP, on page 93, for details on the following features:

- Peer templates
- Route redistribution
- Route maps



CHAPTER 6

# Configuring Advanced BGP

This chapter contains the following sections:

# 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 the Configuring Route Policy Manager section, 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.

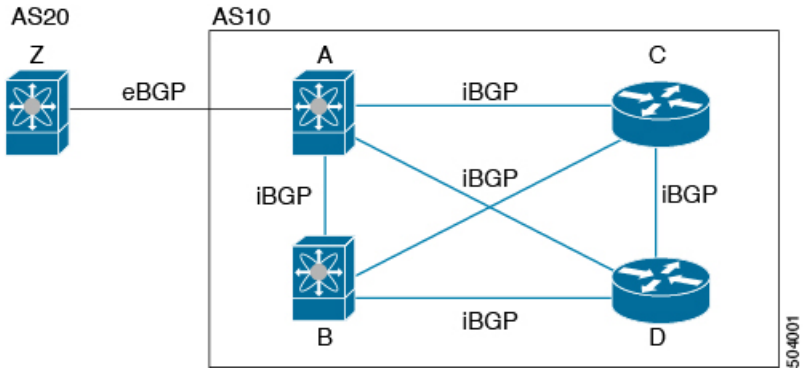
Typically eBGP peerings need to be over directly connected interfaces so that convergence will be faster when the interface goes down.

# 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 10: 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 120 section for information on multihop, fast external fallovers, and limiting the size of the AS_path attribute.

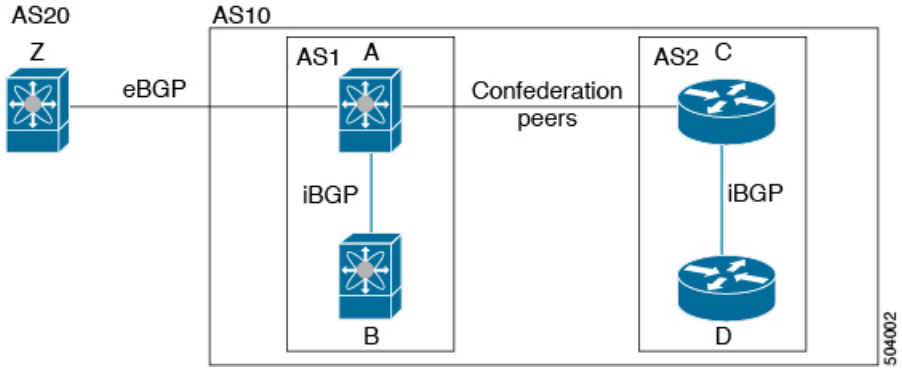
**Note** You should configure a separate interior gateway protocol in the iBGP network.

## 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 11: 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

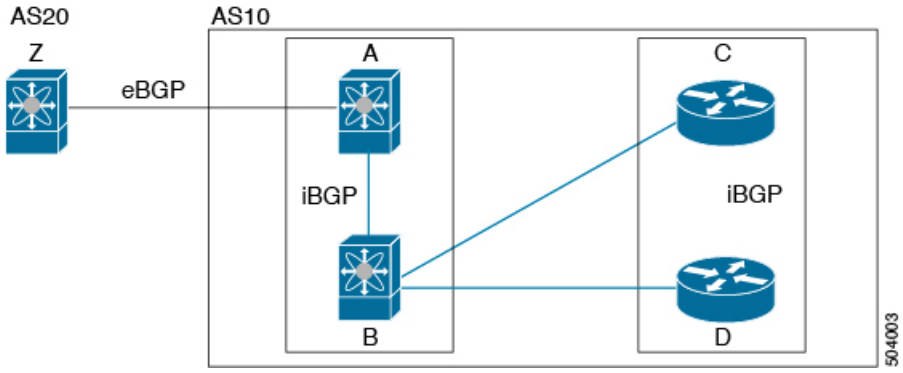
You can alternately reduce the iBGP mesh by using a route reflector configuration where route reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

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 12: 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.

# 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.

# 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.

For information on configuring BGP additional paths, see the Configuring BGP Additional Paths, on page 116 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.

# 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, on page 129 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 recursed metric events trigger a best-path recalculation.

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

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

See the Configuring BGP Next-Hop Address Tracking, on page 113 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 the Configuring Route Policy Manager section, 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
```

# 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.

# 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.

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 , on page 101 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:

- Prefix peering operates only in passive TCP mode. It accepts incoming connections from remote peers if the peer address falls within the prefix.
- Configuring the **advertise-maps** command multiple times is not supported.
- 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.

**Note** Only 48 BGP sessions are validated in Cisco Nexus 3550-T.

- 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.
- The following guidelines and limitations apply to the **remove-private-as** command:
  - It applies only to eBGP peers.
  - It can be configured only in neighbor configuration mode and not in neighbor-address-family mode.
  - If the AS-path includes both private and public AS numbers, the private AS numbers are not removed.
  - If the AS-path contains the AS number of the eBGP neighbor, the private AS numbers are not removed.

- Private AS numbers are removed only if all AS numbers in that AS-path belong to a private AS number range. Private AS numbers are not removed if a peer's AS number or a non-private AS number is found in the AS-path segment.

- If you 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.
- Cisco NX-OS Release 10.2(3t) supports only IPv4 address family.

# 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 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router bgp** *autonomous-system-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Enables BGP and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **template peer-session** *template-name*<br>**Example:**<br>`switch(config-router)# template peer-session BaseSession`<br>`switch(config-router-stmp)#` | Enters peer-session template configuration mode. |
| **Step 4** | (Optional) **password** *number password*<br>**Example:**<br>`switch(config-router-stmp)# password 0 test` | 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** *keepalive hold*<br>**Example:**<br>`switch(config-router-stmp)# timers 30 90` | Adds the BGP keepalive and holdtimer values to the peer-session template.<br>The default keepalive interval is 60. The default hold time is 180. |
| **Step 6** | **exit**<br>**Example:**<br>`switch(config-router-stmp)# exit`<br>`switch(config-router)#` | Exits peer-session template configuration mode. |
| **Step 7** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.168.1.2 remote-as 65535`<br>`switch(config-router-neighbor)#` | Places the router in the neighbor configuration mode for BGP routing and configures the neighbor IP address. |
| **Step 8** | **inherit peer-session** *template-name*<br>**Example:** | Applies a peer-session template to the peer. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router-neighbor)# inherit peer-session BaseSession`<br>`switch(config-router-neighbor)#` | |
| **Step 9** | (Optional) **description** *text*<br>**Example:**<br>`switch(config-router-neighbor)# description Peer Router A`<br>`switch(config-router-neighbor)#` | Adds a description for the neighbor. |
| **Step 10** | (Optional) **show bgp peer-session** *template-name*<br>**Example:**<br>`switch(config-router-neighbor)# show bgp peer-session BaseSession` | Displays the peer-policy template. |
| **Step 11** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor)# copy running-config startup-config` | Saves this configuration change.<br>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® 3550-T 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal` | Enters configuration mode. |
| **Step 2** | **router bgp** *autonomous-system-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Enables BGP and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **template peer-session** *template-name*<br>**Example:**<br>`switch(config-router)# template peer-policy BasePolicy`<br>`switch(config-router-ptmp)#` | Creates a peer-policy template. |
| **Step 4** | (Optional) **advertise-active-only**<br>**Example:**<br>`switch(config-router-ptmp)# advertise-active-only` | Advertises only active routes to the peer. |
| **Step 5** | (Optional) **maximum-prefix** *number*<br>**Example:**<br>`switch(config-router-ptmp)# maximum-prefix 20` | Sets the maximum number of prefixes allowed from this peer. |
| **Step 6** | **exit**<br>**Example:** | Exits peer-policy template configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router-ptmp)# exit`<br>`switch(config-router)#` | |
| **Step 7** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.168.1.2 remote-as 65535`<br>`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** {**ipv4**} {**multicast** \| **unicast**}<br>**Example:**<br>`switch(config-router-neighbor)# address-family ipv4 unicast`<br>`switch(config-router-neighbor-af)#` | Enters global address family configuration mode for the address family specified. |
| **Step 9** | **inherit peer-policy** *template-name preference*<br>**Example:**<br>`switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1` | Applies a peer-policy template to the peer address family configuration and assigns the preference value for this peer policy. |
| **Step 10** | (Optional) **show bgp peer-policy** *template-name*<br>**Example:**<br>`switch(config-router-neighbor-af)# show bgp peer-policy BasePolicy` | Displays the peer-policy template. |
| **Step 11** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor-af)# copy running-config startup-config` | Saves this configuration change.<br>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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal` | Enters global configuration mode. |
| **Step 2** | **router bgp** *autonomous-system-number*<br>**Example:**<br>`switch(config)# router bgp 65535` | Enters BGP mode and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **template peer** *template-name*<br>**Example:**<br>`switch(config-router)# template peer BasePeer` | Enters peer template configuration mode. |
| **Step 4** | (Optional) **inherit peer-session** *template-name*<br>**Example:**<br>`switch(config-router-neighbor)# inherit peer-session BaseSession` | Adds a peer-session template to the peer template. |
| **Step 5** | (Optional) **address-family** {**ipv4**} {**multicast**\|**unicast**}<br>**Example:**<br>`switch(config-router-neighbor)# address-family ipv4 unicast`<br>`switch(config-router-neighbor-af)` | Configures the global address family configuration mode for the specified address family. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | (Optional) **inherit peer-policy** *template-name*<br>**Example:**<br>`switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1` | Applies a peer-policy template to the neighbor address family configuration. |
| **Step 7** | **exit**<br>**Example:**<br>`switch(config-router-neighbor-af)# exit` | Exits BGP neighbor address family configuration mode. |
| **Step 8** | (Optional) **timers** *keepalive hold*<br>**Example:**<br>`switch(config-router-neighbor)# timers 45 100` | Adds the BGP timer values to the peer.<br>These values override the timer values in the peer-session template, BaseSession. |
| **Step 9** | **exit**<br>**Example:**<br>`switch(config-router-neighbor)# exit` | Exits BGP neighbor configuration mode. |
| **Step 10** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.168.1.2 remote-as 65535 switch(config-router-neighbor)#` | Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address. |
| **Step 11** | **inherit peer** *template-name*<br>**Example:**<br>`switch(config-router-neighbor)# inherit peer BasePeer` | Inherits the peer template. |
| **Step 12** | (Optional) **timers** *keepalive hold*<br>**Example:**<br>`switch(config-router-neighbor)# timers 60 120` | Adds the BGP timer values to this neighbor.<br>These values override the timer values in the peer template and the peer-session template. |
| **Step 13** | (Optional) **show bgp peer-template** *template-name*<br>**Example:**<br>`switch(config-router-neighbor)# show bgp peer-template BasePeer` | Displays the peer template. |
| **Step 14** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor)# copy running-config startup-config` | Saves this configuration change.<br>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. 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **timers prefix-peer-timeout** *value*<br>**Example:**<br>`switch(config-router-neighbor)# timers prefix-peer-timeout 120` | Configures the BGP prefix peering timeout value in router configuration mode. The range is from 0 to 1200 seconds. The default value is 30.<br>**Note** For prefix peers, set the prefix peer timeout to be greater than the configured graceful restart timer. If the prefix peer timeout is greater than the graceful restart timer, a peer's route is retained during its restart. If the prefix peer timeout is less than the graceful restart timer, the peer's route is purged by the prefix peer timeout, which may occur before the restart is complete. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **maximum-peers** *value*<br>**Example:**<br>`switch(config-router-neighbor)# maximum-peers 120` | 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 Authentication

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

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **password** {**0** \| **3** \| **7**} *string*<br>**Example:**<br>`switch(config-router-neighbor)# password BGPpassword` | Configures an MD5 password for BGP neighbor sessions. |

# Resetting a BGP Session

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

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

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **soft-reconfiguration inbound**<br>**Example:**<br>`switch(config-router-neighbor-af)# soft-reconfiguration inbound` | Enables soft reconfiguration to store the inbound BGP route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions. |
| **Step 2** | (Optional) **clear bgp** {**ipv4** } {**unicast** \| **multicast** *ip-address* **soft** {**in** \| **out**}<br>**Example:**<br>`switch# clear bgp ip unicast 192.0.2.1 soft in` | Resets the BGP session without tearing down the TCP session. |

# Modifying the Next-Hop Address

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

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

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

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **next-hop-self**<br>**Example:**<br>`switch(config-router-neighbor-af)# next-hop-self` | Uses the local BGP speaker address as the next-hop address in route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions. |
| **Step 2** | **next-hop-third-party**<br>**Example:**<br>`switch(config-router-neighbor-af)# next-hop-third-party` | Sets the next-hop address as a third-party address. Use this command for single-hop eBGP peers that do not have **next-hop-self** configured. |

# Configuring BGP Next-Hop Address Tracking

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

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

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

**Procedure**

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

# Configuring Next-Hop Filtering

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

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

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

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **nexthop route-map** *name*<br>**Example:**<br>`switch(config-router-af)# nexthop route-map nextHopLimits` | Specifies a route map to match the BGP next-hop route to. The name can be any case-sensitive, alphanumeric string up to 63 characters. |

# 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:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | [**no**] **nexthop suppress-default-resolution**<br>**Example:**<br>`switch(config-router)# nexthop suppress-default-resolution` | Prevents resolution of BGP next hop through the IP default route.<br>When this command is enabled:<br>• The output of the **show bgp process detail** command includes the following line: |

| | Command or Action | Purpose |
|---|---|---|
| | | Use default route for nexthop resolution: No<br>• The output of the **show routing clients bgp** command includes the following line:<br>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 |
|---|---|
| **next-hop-self** [all]<br>**Example**:<br>`switch(config-router-af)# next-hop-self all` | Uses the local BGP speaker address as the next-hop address in route updates.<br>The all keyword is optional. If you specify all, all routes are sent to the peer with next-hop-self. If you do not specify all, the next hops of reflected routes are not changed. |

# Shrinking Next-Hop Groups When A Session Goes Down

This feature applies to the following BGP path failure events:

- Any single or multiple Layer 3 link failures
- Line card failures
- 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:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **neighbor-down fib-accelerate**<br>**Example:**<br>`switch(config-router)# neighbor-down fib-accelerate` | Withdraws the corresponding next hop from all next-hop groups (single next-hop routes) whenever a BGP session goes down.<br>**Note** This command applies to both IPv4 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:

**Procedure**

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

# Disabling Policy Batching

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

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

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

**Procedure**

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

# Configuring BGP Additional Paths

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

## 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:

**Procedure**

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

**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:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | [**no**] **additional-paths send**<br>**Example:**<br>`switch(config-router-af)# additional-paths send` | Enables the send capability of additional paths for all of the neighbors under this address family for which the capability has not been disabled.<br>The **no** form of this command disables the send capability. |

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

### 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:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | [**no**] **set ip next-hop unchanged**<br>**Example:**<br>`switch(config-route-map)# set ip next-hop unchanged` | Specifies and unchanged next-hop IP address. |
| **Step 2** | [**no**] **set path-selection all** \| advertise<br>**Example:**<br>`switch(config-route-map)# set path-selection all advertise` | Specifies that all paths be advertised for a given prefix.<br>• all—Advertises all available valid paths.<br>The **no** form of this command specifies that only the best path be advertised. |
| **Step 3** | **show bgp** {**ipv4** } **unicast** [*ip-address*] [**vrf** *vrf-name*]<br>**Example:** | Displays the path ID for the additional paths of a prefix and advertisement information for these paths. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-route-map)# show bgp ipv4 unicast` | |

### 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:

### Procedure

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

### 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:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **disable-connected-check**<br>**Example:**<br>`switch(config-router-neighbor)# disable-connected-check` | Disables checking whether or not a single-hop eBGP peer is directly connected. You must manually reset the BGP sessions after using this command. |

## Configuring eBGP Multihop

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

**Note** This configuration is not supported for BGP interface peering.

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

**Procedure**

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

## Disabling a Fast External Fallover

Be default, the Cisco NX-OS device supports fast external fallover for neighbors in all VRFs and address families (IPv4). 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:

**Procedure**

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

# Limiting the AS-path Attribute

You can configure eBGP to discard routes that have 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:

**Procedure**

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

# Configuring Local AS Support

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

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

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

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **local-as** *number* [**no-prepend** [**replace-as** [**dual-as**]]]<br>**Example:**<br>`switch(config-router-neighbor)# local-as 1.1` | Configures eBGP to prepend the local AS *number* to the AS_PATH attribute. The AS *number* can be a 16-bit integer or a 32-bit integer in the form of a higher 16-bit decimal number and a lower 16-bit decimal number in xx.xx format. |

### 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:

### Procedure

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal` | Enters global configuration mode. |
| **Step 2** | **router bgp** *as-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Enters BGP mode and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **cluster-id** *cluster-id*<br>**Example:**<br>`switch(config-router)# cluster-id 192.0.2.1` | Configures the local router as one of the route reflectors that serve the cluster. You specify a cluster ID to identify the cluster. This command triggers an automatic soft clear or refresh of BGP neighbor sessions. |
| **Step 4** | **address-family** {**ipv4**} {**unicast** \| **multicast**}<br>**Example:**<br>`switch(config-router)# address-family ipv4 unicast`<br>`switch(config-router-af)#` | Enters router address family configuration mode for the specified address family. |
| **Step 5** | (Optional) **client-to-client reflection**<br>**Example:**<br>`switch(config-router-af)# client-to-client reflection` | Configures client-to-client route reflection. This feature is enabled by default. This command triggers an automatic soft clear or refresh of BGP neighbor sessions. |
| **Step 6** | **exit**<br>**Example:**<br>`switch(config-router-af)# exit`<br>`switch(config-router)#` | Exits router address configuration mode. |
| **Step 7** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.0.2.10 remote-as 65535`<br>`switch(config-router-neighbor)#` | Configures the IP address and AS number for a remote BGP peer. |
| **Step 8** | **address-family** {**ipv4**} {**unicast** \| **multicast**}<br>**Example:**<br>`switch(config-router-neighbor)# address-family ipv4 unicast`<br>`switch(config-router-neighbor-af)#` | Enters neighbor address family configuration mode for the unicast IPv4 address family. |
| **Step 9** | **route-reflector-client**<br>**Example:**<br>`switch(config-router-neighbor-af)# route-reflector-client` | Configures the device as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification |

| | Command or Action | Purpose |
|---|---|---|
| | | and session reset for the BGP neighbor sessions. |
| **Step 10** | (Optional) **show bgp** {**ipv4**} {**unicast** \| **multicast**} **neighbors**<br>**Example:**<br>`switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors` | Displays the BGP peers. |
| **Step 11** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor-af)# copy running-config startup-config` | Saves this configuration change. |

### Example

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

```
switch(config)# router bgp 65536
switch(config-router)# neighbor 192.0.2.10 remote-as 65536
switch(config-router-neighbor)# address-family ip unicast
switch(config-router-neighbor-af)# route-reflector-client
switch(config-router-neighbor-af)# copy running-config startup-config
```

# Configuring 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).

You must enter the **set next-hop** command to configure an address family-specific next-hop address.

- 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router bgp** *as-number*<br>**Example:**<br>`switch(config)# router bgp 200`<br>`switch(config-router)#` | Enters BGP mode and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.0.2.12 remote-as 200`<br>`switch(config-router-neighbor)#` | Configures the IP address and AS number for a remote BGP peer. |
| **Step 4** | (Optional) **update-source** *interface number*<br>**Example:**<br>`switch(config-router-neighbor)# update-source loopback 300` | Specifies and updates the source of the BGP session. |
| **Step 5** | **address-family** {**ipv4** } {**unicast** \| **multicast**}<br>**Example:**<br>`switch(config-router-neighbor)# address-family ipv4 unicast`<br>`switch(config-router-neighbor-af)#` | Enters router address family configuration mode for the specified address family. |
| **Step 6** | **route-reflector-client**<br>**Example:**<br>`switch(config-router-neighbor-af)# route-reflector-client` | Configures the device as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification and session reset for the BGP neighbor sessions. |
| **Step 7** | **route-map** *map-name* **out**<br>**Example:**<br>`switch(config-router-neighbor-af)# route-map setrrnh out` | Applies the configured BGP policy to outgoing routes. |
| **Step 8** | (Optional) **show bgp** {**ipv4** } {**unicast** \| **multicast**} [**ip-address**] **route-map** *map-name* [**vrf** *vrf-name*]<br>**Example:**<br>`switch(config-router-neighbor-af)# show bgp ipv4 unicast route-map setrrnh` | Displays the BGP routes that match the route map. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor-af)# copy running-config startup-config` | 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)# 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)# 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
```

# 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:

### Procedure

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

# 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:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **maximum-prefix** *maximum* [*threshold*] [**restart** *time* \| **warning-only**]<br>**Example:**<br>`switch(config-router-neighbor-af)# maximum-prefix 12` | Configures the maximum number of prefixes from a peer. The parameter ranges are as follows:<br>• *maximum*—The range is from 1 to 300000.<br>• *threshold*—The range is from 1 to 100 percent. The default is 75 percent.<br>• *time*—The range is from 1 to 65535 minutes.<br>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.

**Note** Cisco Nexus 3550-T series switch does not honor DSCP or QoS priorities for packets that are forwarded or received on the switch. Below configuration only applies for DSCP settings in packets egressing CPU.

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

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **dscp** *dscp_value*<br>**Example:**<br>`switch(config-router-neighbor)# dscp 63` | 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**, |

| | Command or Action | Purpose |
|---|---|---|
| | Below is an example of the corresponding **show** command:<br>`show ipv4 bgp neighbors`<br>`BGP neighbor is 10.1.1.1, remote AS 0, unknown link, Peer index 4`<br>`BGP version 4, remote router ID 0.0.0.0`<br>`BGP state = Idle, down for 00:13:34, retry in 0.000000`<br>`DSCP (DiffServ CodePoint): 0`<br>`Last read never, hold time = 180, keepalive interval is 60 seconds` | **af21**, **af22**, **af23**, **af31**, **af32**, **af33**, **af41**, **af42**, **af43**, **cs1**, **cs2**, **cs3**, **cs4**, **cs5**, **cs6**, or **cs7**.<br>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:

### Procedure

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

# Configuring Aggregate Addresses

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

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **aggregate-address** *ip-prefix/length* [**as-set**] [**summary-only**] [**advertise-map** *map-name*] [**attribute-map** *map-name*] [**suppress-map** *map-name*]<br>**Example:**<br>`switch(config-router-af)# aggregate-address 192.0.2.0/8 as-set` | Creates an aggregate address. The path advertised for this route is an autonomous system set that consists of all elements contained in all paths that are being summarized:<br>• The **as-set** keyword generates autonomous system set path information and community information from contributing paths. |

| | Command or Action | Purpose |
|---|---|---|
| | | • The **summary-only** keyword filters all more specific routes from updates.<br>• The **advertise-map** keyword and argument specify the route map used to select attribute information from selected routes.<br>• The **attribute-map** keyword and argument specify the route map used to select attribute information from the aggregate. |

# 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:

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **suppress-fib-pending**<br>**Example:**<br>`switch(config-router)# suppress-fib-pending` | Suppresses newly learned BGP routes (IPv4) 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).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters configuration mode. |
| **Step 2** | **router bgp** *as-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Enters BGP mode and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.168.1.2 remote-as 65534`<br>`switch(config-router-neighbor)#` | Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address. |
| **Step 4** | **address-family** {**ipv4**} {**unicast** \| **multicast**}<br>**Example:**<br>`switch(config-router-neighbor)# address-family ipv4 multicast`<br>`switch(config-router-neighbor-af)#` | Enters address family configuration mode. |
| **Step 5** | **advertise-map** *adv-map* {**exist-map** *exist-rmap*\|**non-exist-map** *nonexist-rmap*}<br>**Example:**<br>`switch(config-router-neighbor-af)# advertise-map advertise exist-map exist` | Configures BGP to conditionally advertise routes based on the two configured route maps:<br>• *adv-map*—Specifies a route map with **match** statements that the route must pass before BGP passes the route to the next route map. The *adv-map* is a case-sensitive, alphanumeric string up to 63 characters.<br>• *exist-rmap*—Specifies a route map with match statements for a prefix list. A prefix in the BGP table must match a prefix in the prefix list before BGP advertises the route. The *exist-rmap* is a case-sensitive, alphanumeric string up to 63 characters.<br>• *nonexist-rmap*—Specifies a route map with match statements for a prefix list. A prefix in the BGP table must not match a prefix in the prefix list before BGP advertises the route. The *nonexist-rmap* is a case-sensitive, alphanumeric string up to 63 characters. |

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

### Example

This example shows how to configure BGP conditional advertisement:

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# neighbor 192.0.2.2 remote-as 65537
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# advertise-map advertise exist-map exist
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# exit
switch(config-router)# exit
switch(config)# route-map advertise
switch(config-route-map)# match as-path pathList
switch(config-route-map)# exit
switch(config)# route-map exit
switch(config-route-map)# match ip address prefix-list plist
switch(config-route-map)# exit
switch(config)# ip prefix-list plist permit 209.165.201.0/27
```

# Configuring Route Redistribution

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

### Procedure

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

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# router bgp 65535`<br>`switch(config-router)#` | |
| **Step 3** | **address-family ipv4** {**unicast** \| **multicast**}<br>**Example:**<br>`switch(config-router)# address-family ipv4 unicast`<br>`switch(config-router-af)#` | Enters address family configuration mode. |
| **Step 4** | **redistribute** {**direct**\| {**eigrp** \|**ospf** \|**rip**} *instance-tag* \| **static**} **route-map** *map-name*<br>**Example:**<br>`switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap` | Redistributes routes from other protocols into BGP. |
| **Step 5** | (Optional) **default-metric** *value*<br>**Example:**<br>`switch(config-router-af)# default-metric 33` | Generates a default route into BGP. |
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`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
```

# 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).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **route-map allow permit**<br>**Example:**<br>`switch(config)# route-map allow permit`<br>`switch(config-route-map)#` | Enters router map configuration mode and defines the conditions for redistributing routes. |
| **Step 3** | **exit**<br>**Example:**<br>`switch(config-route-map)# exit`<br>`switch(config)#` | Exits router map configuration mode. |
| **Step 4** | **ip route** *ip-address network-mask* **null** *null-interface-number*<br>**Example:**<br>`switch(config)# ip route 192.0.2.1 255.255.255.0 null 0` | Configures the IP address. |
| **Step 5** | **router bgp** *as-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Enters BGP mode and assigns the AS number to the local BGP speaker. |
| **Step 6** | **address-family** {**ipv4**} **unicast**<br>**Example:**<br>`switch(config-router)# address-family ipv4 unicast`<br>`switch(config-router-af)#` | Enters address-family configuration mode. |
| **Step 7** | **default-information originate**<br>**Example:**<br>`switch(config-router-af)# default-information originate` | Advertises the default route. |
| **Step 8** | **redistribute static route-map allow**<br>**Example:**<br>`switch(config-router-af)# redistribute static route-map allow` | Redistributes the default route. |
| **Step 9** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-af)# copy running-config startup-config` | Saves this configuration change. |

# Configuring BGP Attribute Filtering and Error Handling

You can configure BGP attribute filtering and error handling to provide an increased level of security. The following features are available and implemented in the following order:

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

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

## Treating as Withdraw Path Attributes from a BGP Update Message

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

**Procedure**

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

## Discarding Path Attributes from a BGP Update Message

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

**Procedure**

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

# Enabling or Disabling Enhanced Attribute Error Handling

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

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

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | [**no**] **enhanced-error**<br>**Example:**<br>`switch(config)# router bgp 1000`<br>`switch(config-router)# enhanced-error` | Enables or disables BGP enhanced attribute error handling. |

# Displaying Discarded or Unknown Path Attributes

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

| Command | Purpose |
|---|---|
| **show bgp** {**ipv4** } **unicast path-attribute discard**] | Displays all prefixes for which an attribute has been discarded. |
| **show bgp** {**ipv4** } **unicast path-attribute unknown**] | Displays all prefixes that have an unknown attribute. |
| **show bgp** {**ipv4** } **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 |
| --- | --- |
| **bestpath** [**always-compare-med** \| **as-pathmultipath-relax** \| **compare-routerid** \|**cost-community ignore** \| \|**med** {**confed** \|**missing-as-worst**\| **non-deterministic**}]<br>**Example:**<br>`switch(config-router)# bestpath always-compare-med` | Modifies the best-path algorithm. The optional parameters are as follows:<br>• **always-compare-med** —Compares MED on paths from different autonomous systems.<br>• **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.<br>• **compare-routerid** —Compares the router IDs for identical eBGP paths.<br>• **cost-community ignore** —Ignores the cost community for BGP best-path calculations.<br>• **med confed** —Forces bestpath to do a MED comparison only between paths originated within a confederation.<br>• **med missing-as-worst** —Treats a missing MED as the highest MED.<br>• **med non-deterministic** —Does not always pick the best MED path from among the paths from the same autonomous system. |
| **enforce-first-as**<br>**Example:**<br>`switch(config-router)# enforce-first-as` | Enforces the neighbor autonomous system to be the first AS number listed in the AS_path attribute for eBGP. |
| **log-neighbor-changes**<br>**Example:**<br>`switch(config-router)# log-neighbor-changes` | Generates a system message when any neighbor changes state.<br>**Note** To suppress neighbor status change messages for a specific neighbor, you can use the **log-neighbor-changes disable** command in router address-family configuration mode. |
| **router-id** *id*<br>**Example:**<br>`switch(config-router)# router-id 10.165.20.1` | Manually configures the router ID for this BGP speaker. |

| Command | Purpose |
| --- | --- |
| **timers** [**bestpath-delay** *delay* \| *bgpkeepalive holdtime* \| **prefix-peer-timeout** *timeout*]<br>**Example:**<br>`switch(config-router)# timers bgp 90 270` | Sets BGP timer values. The optional parameters are as follows:<br>• *delay* —Initial best-path timeout value after a restart. The range is from 0 to 3600 seconds. The default value is 300.<br>• *keepalive* —BGP session keepalive time. The range is from 0 to 3600 seconds. The default value is 60.<br>• *holdtime* —BGP session hold time. The range is from 0 to 3600 seconds. The default value is180.<br>• *timeout* —Prefix peer timeout value. The range is from 0 to 1200 seconds. The default value is 30.<br>You must manually reset the BGP sessions after configuring this command. |

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

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

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

| Command | Purpose |
|---|---|
| **description** *string*<br><br>**Example:**<br><br>`switch(config-router-neighbor)# description main site` | Sets a descriptive string for this BGP peer. The string can be up to 80 alphanumeric characters. |
| **low-memory exempt**<br><br>**Example:**<br><br>`switch(config-router-neighbor)# low-memory exempt` | Exempts this BGP neighbor from a possible shutdown due to a low memory condition. |
| **transport connection-mode passive**<br><br>**Example:**<br><br>`switch(config-router-neighbor)# transport connection-mode passive` | Allows a passive connection setup only. This BGP speaker does not initiate a TCP connection to a BGP peer. You must manually reset the BGP sessions after configuring this command. |
| [**no** \| **default**] **remove-private-as** [**all** \| **replace-as**]<br><br>**Example:**<br><br>`switch(config-router-neighbor)# remove-private-as` | Removes private AS numbers from outbound route updates to an eBGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.<br><br>The optional parameters are as follows:<br>• **no** —Disables the command.<br>• **default** —Moves the command to its default mode.<br>• **all** —Removes all private-as numbers from the AS-path value.<br>• **replace-as** —Replaces all private AS numbers with the replace-as AS-path value.<br><br>See the Guidelines and Limitations for Advanced BGP, on page 103 section for additional information on this command. |
| **update-source** *interface-type number*<br><br>**Example:**<br><br>`switch(config-router-neighbor)# update-source ethernet 1/1` | Configures the BGP speaker to use the source IP address of the configured interface for BGP sessions to the peer. This command triggers an automatic notification and session reset for the BGP neighbor sessions. Single-hop iBGP peers support fast external fallover when **update-source** is configured. |

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

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

| Command | Purpose |
|---|---|
| **[no \| default] as-override**<br>**Example:**<br>`switch(config-router-neighbor-af)# as-override` | **no** - (Optional) Disables the command.<br>**default** - (Optional) Moves the command to its default mode.<br>**as-override** - While sending updates to eBGP peer, replaces in the *path* attribute all occurrences of the peer's AS number with the local AS number. |

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | | |

# 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | switch(config)# **ip prefix-list** *name* **seq** *number* **permit** *prefix-length* | Creates a prefix list to match IP packets or routes with the permit keyword. |
| **Step 3** | switch(config)# **route-map** *map-tag* **permit** *sequence-number* | 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** *prefix-list-name* | 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** *value1 value2 value3* | Specifies the administrative distance for interior BGP (iBGP) or exterior BGP (eBGP) |

| | Command or Action | Purpose |
|---|---|---|
| | | routes and BGP routes originated in the local autonomous system. The range is from 1 to 255.<br>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** *as-number* | Enters BGP mode and assigns the AS number to the local BGP speaker. |
| **Step 8** | switch(config-router)# **address-family** {**ipv4** \| **vpnv4**} **unicast** | Enters address family configuration mode. |
| **Step 9** | switch(config-router-af)# **table-map** *map-name* | 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.<br>**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 unicast and multicast routes.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:** | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch# configure terminal`<br>`switch(config)#` | |
| **Step 2** | **router bgp** *as-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Enters BGP mode and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **neighbor** *ip-address* **remote-as** *as-number*<br>**Example:**<br>`switch(config-router)# neighbor 192.168.1.2 remote-as 65534`<br>`switch(config-router-neighbor)#` | Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor-af)# copy running-config startup-config` | Saves this configuration change. |

### Example

# Configuring BMP

You can configure BMP on the Cisco Nexus® 3550-T device.

### Before you begin

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **router bgp as-number**<br>**Example:**<br>`switch(config)# router bgp 200`<br>`switch(config-router)#` | Enters BGP mode and assigns the autonomous system number to the local BGP speaker. |
| **Step 3** | **bmp server** *server-number*<br>**Example:**<br>`switch(config-router)# bmp server 1` | Configures the BMP server to which BGP should send information. The server number is used as a key. |

| | Command or Action | Purpose |
|---|---|---|
| | | **Note** You can configure up to two BMP servers. |
| **Step 4** | **address ip-address port-number port-number**<br>**Example:**<br>`switch(config-router)# address 10.1.1.1 port-number 2000` | Configures the IPv4 address of the host and the port number on which the BMP speaker connects to the BMP server. |
| **Step 5** | **description** *string*<br>**Example:**<br>`switch(config-router)# description BMPserver1` | Configures the BMP server description. You can enter up to 256 alphanumeric characters. |
| **Step 6** | **initial-refresh** { *skip* \| *delay time*}<br>**Example:**<br>`switch(config-router)# initial-refresh delay 100` | Configures the option to send a route refresh when BGP is converged and the BMP server connection is established later.<br>The skip option specifies to not send a route refresh if the BMP server connection comes up later.<br>The delay option specifies the time in seconds after which the route refresh should be sent. The range is from 30 to 720 seconds, and the default value is 30 seconds. |
| **Step 7** | **initial-delay** *time*<br>**Example:**<br>`switch(config-router)# initial-delay 120` | Configures the delay after which a connection is attempted to the BMP server. The range is from 30 to 720 seconds, and the default value is 45 seconds. |
| **Step 8** | **stats-reporting-period** *time*<br>**Example:**<br>`switch(config-router)# stats-reporting-period 50` | 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<br>**Example:**<br>`switch(config-router)# shutdown` | Disables the connection to the BMP server. |
| **Step 10** | **neighbor ip-address**<br>**Example:**<br>`switch(config-router)# neighbor 192.168.1.2`<br>`switch(config-router-neighbor)#` | Enters neighbor configuration mode for BGP routing and configures the neighbor IP address. |
| **Step 11** | **remote-as** *as-number*<br>**Example:** | Configures the AS number for a remote BGP peer. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-router-neighbor)# remote-as 65535` | |
| **Step 12** | **bmp-activate-server** *server-number*<br>**Example:**<br>`switch(config-router-neighbor)# bmp-activate-server 1` | Configures the BMP server to which a neighbor's information should be sent. |
| **Step 13** | (Optional) **show bgp bmp** *server [server-number] [detail]*<br>**Example:**<br>`switch(config-router-neighbor)# show bgp bmp server` | Displays BMP server information. |
| **Step 14** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router-neighbor)# copy running-config startup-config` | Saves this configuration change. |

# About BGP Graceful Shutdown

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 146. 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***]**<br>**graceful-shutdown aware** |
| Peer | A BGP peer or a link between neighbors. For example, advertise only one link between peers with GRACEFUL_SHUTDOWN community. | **graceful-shutdown activate [route-map** *route-map***]** |

# Graceful Shutdown with Route Maps

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

### Normal Inbound Route Maps

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

Cisco Nexus® switches do not require an inbound route map for the graceful shutdown feature. Cisco NX-OS switch 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 that do not have the implicit inbound route map, a graceful shutdown inbound route map to use the graceful shutdown feature on these switches. 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 compatible version of NX-OS and is receiving the graceful shutdown routes.
- If you want to disable graceful shutdown aware, but still want the router to act on incoming routes with GRACEFUL_SHUTDOWN community from some BGP neighbors, you can configure an inbound route map under the respective peers.

### Normal Outbound Route Maps

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

### Graceful Shutdown Outbound Route Maps

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

You can use outbound route maps in the following ways:

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

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

### Route Map Precedence

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

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

# Guidelines and Limitations

The following are limitations and guidelines for BGP global shutdown:

- Graceful shutdown feature can only help avoid traffic loss when alternative routes exist in the network for the affected routers. If the router has no alternate routes, routes carrying the GRACEFUL_SHUTDOWN community are the only ones available, and therefore, are used in the best-path calculation. This situation defeats the purpose of the feature.
- Configuring a BGP send community is required to send the GRACEFUL_SHUTDOWN community.
- For route maps:
  - When global route maps and neighbor route maps are configured, the per-neighbor route maps take precedence.
  - Outbound route maps take precedence over any global route maps configured for graceful shutdown.
  - Outbound route maps take precedence over any peer route maps configured for graceful shutdown.
  - To add the graceful shutdown functionality to legacy (existing) inbound route maps, follow this order:
    1. Add the graceful shutdown match clause to the top of the route map by setting a low sequence number for the clause (for example, sequence number 0).
    2. Add a continue statement after the graceful shutdown clause. If you omit the continue statement, route-map processing stops when it matches the graceful shutdown clause, any other clauses with higher sequence numbers (for example, 1 and higher) are not processed.

# Graceful Shutdown Task Overview

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

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

To bring the router or link down:

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

To bring the router or link back online:

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

# Configuring Graceful Shutdown on a Link

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

### Before you begin

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **config terminal**<br>**Example:**<br>`switch-1# configure terminal`<br>`switch-1(config)#` | Enters global configuration mode. |
| **Step 2** | **router bgp** *autonomous-system-number*<br>**Example:**<br>`switch-1(config)# router bgp 110`<br>`switch-1(config-router)#` | Enters router configuration mode to create or configure a BGP routing process. |
| **Step 3** | **neighbor {** *ipv4-address*} **remote-as** *as-number*<br>**Example:**<br>`switch-1(config-router)# neighbor 10.0.0.3 remote-as 200`<br>`switch-1(config-router-neighbor)#` | Configures the autonomous system (AS) to which the neighbor belongs. |
| **Step 4** | **graceful-shutdown activate [route-map** *map-name***]**<br>**Example:**<br>`switch-1(config-router-neighbor)# graceful-shutdown activate route-map gshutPeer out`<br>`switch-1(config-router-neighbor)#` | Configures graceful shutdown on the link to the neighbor. Also, advertises the routes with the well-known GRACEFUL_SHUTDOWN community and applies the route map to the outbound route updates.<br>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.<br>The devices receiving the gshut community look at the communities of the route and optionally use the communities to apply routing policy. |

# Filtering BGP Routes and Setting Local Preference Based On GRACEFUL_SHUTDOWN Communities

Switches do not have an inbound route map that matches against the GRACEFUL_SHUTDOWN community name. Therefore, they have no way of identifying and depreferring the correct routes.

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

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch-1# configure terminal`<br>`switch-1<config)#` | Enters global configuration mode. |
| **Step 2** | **ip community list standard** *community-list-name* **seq** *sequence-number* **{ permit \| deny }** *value*<br>**Example:**<br>`switch-1(config)# ip community-list standard GSHUT seq 10 permit 65535:0`<br>`switch-1(config)#` | Configures a community list and permits or denies routes that have the well-known graceful shutdown community value. |
| **Step 3** | **route map** *map-tag* **{deny \| permit}** *sequence-number*<br>**Example:**<br>`switch-1(config)# route-map RM_GSHUT permit 10`<br>`switch-1(config-route-map)#` | Configures a route map as sequence 10 and permits routes that have the GRACEFUL_SHUTDOWN community. |
| **Step 4** | **match community** *community-list-name*<br>**Example:**<br>`switch-1(config-route-map)# match community GSHUT`<br>`switch-1(config-route-map)#` | Configures that routes that match the IP community list GSHUT are processed by Route Policy Manager (RPM). |
| **Step 5** | **set local-preference** *local-pref-value*<br>**Example:**<br>`switch-1(config-route-map)# set local-preference 10`<br>`switch-1(config-route-map)#` | Configures that the routes that match the IP community list GSHUT will be given a specified local preference. |
| **Step 6** | **exit**<br>**Example:** | Leaves route map configuration and returns to global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch-1(config-route-map)# exit`<br>`switch-1(config)#` | |
| **Step 7** | **router bgp** *community-list-name*<br>**Example:**<br>`switch-1(config)# router bgp 100`<br>`switch-1(config-router)#` | Enters router configuration mode and creates a BGP instance. |
| **Step 8** | **neighbor {** *ipv4-address* **}**<br>**Example:**<br>`switch-1(config-router)# neighbor 10.0.0.3`<br>`switch-1(config-router-neighbor)#` | Enters route BGP neighbor mode for a specified neighbor. |
| **Step 9** | **address-family {** *address-family sub family* **}**<br>**Example:**<br>`nxosv2(config-router-neighbor)# address-family ipv4 unicast`<br>`nxosv2(config-router-neighbor-af)#` | Puts the neighbor into address family (AF) configuration mode. |
| **Step 10** | **send community**<br>**Example:**<br>`nxosv2(config-router-neighbor-af)# send-community`<br>`nxosv2(config-router-neighbor-af)#` | Enables BGP community exchange with the neighbor. |
| **Step 11** | **route map** *map-tag* **in**<br>**Example:**<br>`nxosv2(config-router-neighbor-af)# route-map RM_GSHUT in`<br>`nxosv2(config-router-neighbor-af)#` | Applies the route map to incoming routes from the neighbor. In this example, the route map that is named RM_GSHUT permits routes with the GRACEFUL_SHUTDOWN community from the neighbor. |

# Configuring Graceful Shutdown for All BGP Neighbors

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

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

### Before you begin

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

### Procedure

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

# Controlling the Preference for All Routes with the GRACEFUL_SHUTDOWN Community

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

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

### Before you begin

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

**Procedure**

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

# Preventing Sending the GRACEFUL_SHUTDOWN Community to a Peer

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

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

Choose either of the following methods:

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | Disable the send-community in the running config.`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. |
| **Step 2** | Delete the GRACEFUL_SHUTDOWN community through an outbound route map by following these steps: | If you use this option, the community list matches and permits the GRACEFUL_SHUTDOWN community, then the outbound route map matches against the |

| | Command or Action | Purpose |
|---|---|---|
| | | community and then deletes it from the outbound route map. All other communities pass through the outbound route map without issue. |

# Displaying Graceful Shutdown Information

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

| Command | Action |
|---|---|
| **show ip bgp community-list graceful-shutdown** | Shows all entires in the BGP routing table that have the GRACEFUL_SHUTDOWN community. |
| **show running-config bgp** | Shows the running BGP configuration. |
| **show running-config bgp all** | Shows all information for the running BGP configuration including information about the graceful shutdown feature. |
| **show bgp** *address-family* **neighbors** *neighbor-address* | When the feature is configured for the peer, shows the following:<br>• The state of the graceful-shutdown-activate feature for the specified neighbor<br>• The name of any graceful shutdown route map configured for the specified neighbor |
| **show bgp process** | Shows different information depending on the context.<br>When the graceful-shutdown-activate option is configured in peer context, shows the enabled or disabled state for the feature through `graceful-shutdown-active`.<br>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:<br>• `graceful-shutdown-active`<br>• `graceful-shutdown-aware`<br>• `graceful-shutdown route-map` |

| Command | Action |
|---|---|
| **show ip bgp** *address* | For the specified address, shows the BGP routing table information, including the following:<br>• The state of the specified address as the best path<br>• 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.

### Before you begin

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

Create the VRFs.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters configuration mode. |
| **Step 2** | **router bgp** *as-number*<br>**Example:**<br>`switch(config)# router bgp 65535`<br>`switch(config-router)#` | Creates a new BGP process with the configured autonomous system number. |
| **Step 3** | (Optional) **timers prefix-peer-timeout** *timeout*<br>**Example:**<br>`switch(config-router)# timers prefix-peer-timeout 20` | Configures the timeout value (in seconds) for BGP prefix peers. The default value is 90 seconds.<br>**Note** This command is supported beginning with Cisco NX-OS Release 9.3(3). |
| **Step 4** | **graceful-restart**<br>**Example:**<br>`switch(config-router)# graceful-restart` | Enables a graceful restart and the graceful restart helper functionality. This command is enabled by default.<br>This command triggers an automatic notification and session reset for the BGP neighbor sessions. |
| **Step 5** | **graceful-restart** {**restart-time** *time*\|**stalepath-time** *time*}<br>**Example:**<br>`switch(config-router)# graceful-restart restart-time 300` | Configures the graceful restart timers.<br>The optional parameters are as follows:<br>• **restart-time**—Maximum time for a restart sent to the BGP peer. The range is from 1 to 3600 seconds. The default is 120.<br>• **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 |
|---|---|---|
| | | This command triggers an automatic notification and session reset for the BGP neighbor sessions. |
| **Step 6** | **graceful-restart-helper**<br>**Example:**<br>`switch(config-router)# graceful-restart restart-time 300` | Enables the graceful restart helper functionality. Use this command if you have disabled graceful restart but you still want to enable graceful restart helper functionality. This command triggers an automatic notification and session reset for the BGP neighbor sessions. |
| **Step 7** | (Optional) **show running-config bgp**<br>**Example:**<br>`switch(config-router)# show running-config bgp` | Displays the BGP configuration. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-router)# copy running-config startup-config` | 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.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal switch(config)#` | Enters global configuration mode. |

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

### Example

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

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# exit
switch(config)# router bgp 65536
switch(config-router)# vrf NewVRF
switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65536
switch(config-router-vrf-neighbor)# copy running-config startup-config
```

# Verifying the Advanced BGP Configuration

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

| Command | Purpose |
|---|---|
| **show bgp all** [**summary**] [**vrf** *vrf-name*] | Displays the BGP information for all address families. |
| **show bgp convergence [vrf** *vrf-name]* | Displays the BGP information for all address families. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address* ] **community** {**regexp** *expression* \| [**community**] [**no-advertise**] [**no-export**] [**no-export-subconfed**]} [**vrf** *vrf-name*] | Displays the BGP routes that match a BGP community. |
| **show bgp** [**vrf** *vrf-name*] {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **community-list** *list-name* [**vrf** *vrf-name*] | Displays the BGP routes that match a BGP community list. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **extcommunity** {**regexp** *expression* \| **generic** [**non-transitive** \| **transitive**] *aa4:nn* [**exact-match**]} [**vrf** *vrf-name*] | Displays the BGP routes that match a BGP extended community. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **extcommunity-list** *list-name* [**exact-match**]} [**vrf** vrf-name] | Displays the BGP routes that match a BGP extended community list. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **extcommunity-list** *list-name* [**exact-match**]} [**vrf** vrf-name] | Displays the information for BGP route dampening. Use the **clear bgp dampening** command to clear the route flap dampening information. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] {**dampening dampened-paths** [**regexp** *expression*]} [**vrf** vrf-name] | Displays the BGP route history paths. |
| **show bgp** {**ipv4** \| **vpnv4**} {**unicast** \| **multicast**} [*ip-address*] **filter-list** *list-name* [**vrf** *vrf-name*] | Displays the information for the BGP filter list. |
| **show bgp** {**ipv4** \| **vpnv4**} {**unicast** \| **multicast**} [*ip-address*] **neighbors** [*ip-address*] [**vrf** *vrf-name*] | Displays the information for BGP peers. Use the **clear bgp neighbors** command to clear these neighbors. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] {**nexthop** \| **nexthop-database**} [**vrf** *vrf-name*] | Displays the information for the BGP route next hop. |
| **show bgp paths** | Displays the BGP path information. |
| **show bgp** {**ipv4**} {**unicast** \| **multicast**} [*ip-address*] **policy** *name* [**vrf** *vrf-name*] | Displays the BGP policy information. Use the **clear bgp policy** command to clear the policy information. |

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

# Monitoring BGP Statistics

To display BGP statistics, use the following commands:

| Command | Purpose |
|---|---|

| | |
|---|---|
| **show bgp** {**ipv4** } {**unicast** } [*ip-address*] **flap-statistics** [**vrf** *vrf-name*] | Displays the BGP route flap statistics. Use the **clear bgp flap-statistics** command to clear these statistics. |
| **show bgp** {**ipv4**} **unicast injected-routes** | Displays injected routes in the routing table. |
| **show bgp sessions** [**vrf** *vrf-name*] | Displays the BGP sessions for all peers. Use the **clear bgp sessions** command to clear these statistics. |
| **show bgp statistics** | Displays the BGP statistics. |

# Configuration Examples

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

```
template peer BasePeer-V4
    description BasePeer-V4
    password 3 f4200cfc725bbd28
    address-family ipv4 unicast
--
      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
```

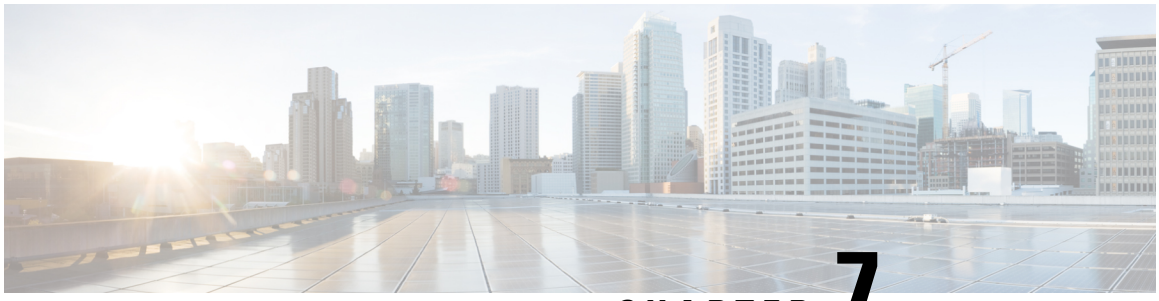
# Related Topics

The following topics can give more information on BGP:

- Configuring Basic BGP, on page 75
- Configuring Route Policy Manager section

# Additional References

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



CHAPTER 7

# Configuring Static Routing

This chapter describes how to configure static routing on the Cisco NX-OS device.

This chapter contains the following sections:



# About Static Routing

Routers forward packets using either route information from route table entries that you manually configure or the route information that is calculated using dynamic routing algorithms.

Static routes, which define explicit paths between two routers, cannot be automatically updated. You must manually reconfigure static routes when network changes occur. Static routes use less bandwidth than dynamic routes. No CPU cycles are used to calculate and analyze routing updates.

You can supplement dynamic routes with static routes where appropriate. You can redistribute static routes into dynamic routing algorithms, but you cannot redistribute routing information calculated by dynamic routing algorithms into the static routing table.

You should use static routes in environments where network traffic is predictable and where the network design is simple. You should not use static routes in large, constantly changing networks because static routes cannot react to network changes. Most networks use dynamic routes to communicate between routers but might have one or two static routes configured for special cases. Static routes are also useful for specifying a gateway of last resort (a default router to which all unroutable packets are sent).

## Administrative Distance

An administrative distance is the metric used by routers to choose the best path when there are two or more routes to the same destination from two different routing protocols. An administrative distance guides the selection of one routing protocol (or static route) over another, when more than one protocol adds the same route to the unicast routing table. Each routing protocol is prioritized in order of most to least reliable using an administrative distance value.

Static routes have a default administrative distance of 1. A router prefers a static route to a dynamic route because the router considers a route with a low number to be the shortest. If you want a dynamic route to override a static route, you can specify an administrative distance for the static route. For example, if you have two dynamic routes with an administrative distance of 120, you would specify an administrative distance that is greater than 120 for the static route if you want the dynamic route to override the static route.

## Directly Connected Static Routes

You must specify only the output interface (the interface on which all packets are sent to the destination network) in a directly connected static route. The router assumes that the destination is directly attached to the output interface and the packet destination is used as the next-hop address. The next hop can be an interface, but only for point-to-point interfaces. For broadcast interfaces, the next hop must be an IPv4 address.

## Fully Specified Static Routes

You must specify either the output interface (the interface on which all packets are sent to the destination network) or the next-hop address in a fully specified static route. You can use a fully specified static route when the output interface is a multi-access interface and you need to identify the next-hop address. The next-hop address must be directly attached to the specified output interface.

## Floating Static Routes

A floating static route is a static route that the router uses to back up a dynamic route. You must configure a floating static route with a higher administrative distance than the dynamic route that it backs up. In this instance, the router prefers a dynamic route to a floating static route. You can use a floating static route as a replacement if the dynamic route is lost.

**Note** By default, a router prefers a static route to a dynamic route because a static route has a smaller administrative distance than a dynamic route.

## Remote Next Hops for Static Routes

You can specify the next-hop address of a neighboring router that is not directly connected to the router for static routes with remote (non-directly attached) next hops. If a static route has remote next hops during data forwarding, the next hops are recursively used in the unicast routing table to identify the corresponding directly attached next hops that have reachability to the remote next hops.

## Virtualization Support

Static routes support virtual routing and forwarding (VRF) instances.

# Prerequisites for Static Routing

Static routing has the following prerequisites:

- If the next-hop address for a static route is unreachable, the static route is not added to the unicast routing table.

# Default Settings

The table lists the default settings for static routing parameters.

*Table 6: Default Static Routing Parameters*

| Parameters | Default |
|---|---|
| Administrative distance | 1 |
| RIP feature | Disabled |

# Configuring Static Routing

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

## Configuring a Static Route

You can configure a static route on the device.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | Enter the following command:<br>**Example:**<br>`switch(config)# ip route 192.0.2.0/8 ethernet 1/2 192.0.2.4` | **ip route** {*ip-prefix* \| *ip-addr*/*ip-mask*} {[*next-hop* \| *nh-prefix*] \| [*interface next-hop* \| *nh-prefix*]} [**name** *nexthop-name*] [**tag** *tag-value*] [*preference*]<br>Configures a static route and the interface for this static route. Use **?** to display a list of supported interfaces. You can specify a null interface by using **null 0**.<br>You can optionally configure the next-hop address. |

| | Command or Action | Purpose |
|---|---|---|
| | | The *preference* value sets the administrative distance. The range is from 1 to 255. The default is 1.<br>**Note** Use the **no** {**ip**} **route** command to remove the static route. |
| **Step 3** | (Optional) **show** {**ip**} **static-route**<br>**Example:**<br>`switch(config)# show ip static-route` | Displays information about static routes. |
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to configure a static route for a null interface:

```
switch# configure terminal
switch(config)# ip route 1.1.1.1/32 null 0
switch(config)# copy running-config startup-config
```

# Configuring a Static Route Over a VLAN

You can configure a static route without next-hop support over a VLAN.

### Before you begin

Ensure that the access port is part of the VLAN.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **feature interface vlan**<br>**Example:**<br>`switch(config)# feature interface-vlan` | Enables VLAN interface mode. |
| **Step 3** | **interface-vlan** *vlan-id*<br>**Example:** | Creates an SVI and enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# interface-vlan 10` | The range for the **vlan-id** argument is from 1 to 4094, except for the VLANs reserved for the internal switch. |
| **Step 4** | **ip address** *ip-addr*/*length*<br>**Example:**<br>`switch(config)# ip address 192.0.2.1/8` | Configures an IP address for the VLAN. |
| **Step 5** | [**no**] **ip route** *ip-addr*/*length vlan-id*<br>**Example:**<br>`switch(config)# ip route 209.165.200.224/27 vlan 10` | Adds an interface static route without a next hop on the switch virtual interface (SVI).<br>The IP address is the address that is configured on the interface that is connected to the switch.<br>Use the **no** keyword with this command to remove the static route. |
| **Step 6** | (Optional) **show ip route**<br>**Example:**<br>`switch(config)# show ip route` | Displays routes from the Unicast Route Information Base (URIB). |
| **Step 7** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to configure a static route without a next hop over an SVI:

```
switch# configure terminal
switch(config)# feature interface-vlan
swicth(config)# interface vlan 10
switch(config-if)# ip address 192.0.2.1/8
switch(config-if)# ip route 209.165.200.224/27 vlan 10  <===209,165.200.224 is the IP
address of the interface that is configured on the interface that is directly connected to
the switch.
switch(config-if)# copy running-config startup-config
```

# Configuring Virtualization

You can configure a static route in a VRF.

**Note** When a **ip route** command is applied on a VRF context, the **show run vrf** command displays some octets that have changed from the initial configuration.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **vrf context** *vrf-name*<br>**Example:**<br>`switch(config)# vrf context StaticVrf`<br>`switch(config-vrf)#` | Creates a VRF and enters VRF configuration mode. |
| **Step 3** | **ip route** {*ip-prefix* \| *ip-addr ip-mask*} {*next-hop* \| *nh-prefix* \| *interface*} [**name** *nexthop-name*] [**tag** *tag-value*] [*preference*]<br>**Example:**<br>`switch(config-vrf)# ip route 192.0.2.0/8 ethernet 1/2` | Configures a static route and the interface for this static route. Use **?** to display a list of supported interfaces. You can specify a null interface by using **null 0**.<br>You can optionally configure the next-hop address.<br>The *preference* value sets the administrative distance. The range is from 1 through 255. The default is 1. |
| **Step 4** | (Optional) **show** {**ip**} **static-route vrf** *vrf-name*<br>**Example:**<br>`switch(config-vrf)# show ip static-route` | Displays information about static routes. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-vrf)# copy running-config startup-config` | Saves this configuration change. |

### Example

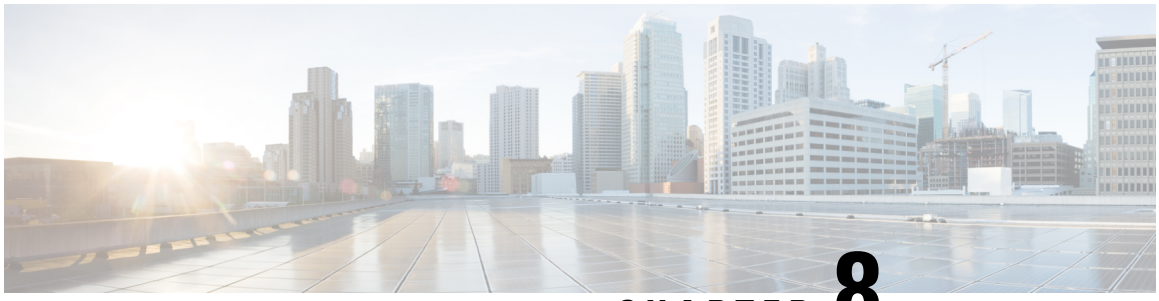
This example shows how to configure a static route:

```
switch# configure terminal
switch(config)# vrf context StaticVrf
switch(config-vrf)# ip route 192.0.2.0/8 192.0.2.10
switch(config-vrf)# copy running-config startup-config
```

# Configuration Example for Static Routing

This example shows how to configure static routing:

```
configure terminal
ip route 192.0.2.0/8 192.0.2.10
copy running-config startup-config
```



CHAPTER 8

# Configuring Layer 3 Virtualization

This chapter describes how to configure Layer 3 virtualization on the Cisco NX-OS device.

This chapter includes the following sections:



## About Layer 3 Virtualization

Cisco NX-OS supports multiple virtual routing and forwarding instances (VRFs). Each VRF contains a separate address space with unicast and multicast route tables for IPv4 and makes routing decisions independent of any other VRF.

Each router has a default VRF and a management VRF.

Management VRF

- The management VRF is for management purposes only.
- Only the mgmt 0 interface can be in the management VRF.
- The mgmt 0 interface cannot be assigned to another VRF.
- No routing protocols can run in the management VRF (static only).

Default VRF

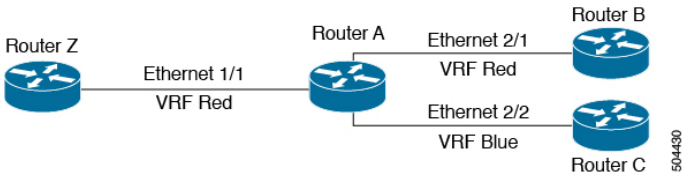
- All Layer 3 interfaces exist in the default VRF.
- Routing protocols run in the default VRF context.
- The default VRF uses the default routing context for all show commands.
- The default VRF is similar to the global routing table concept in Cisco IOS.

# VRF and Routing

All unicast and multicast routing protocols support VRFs. When you configure a routing protocol in a VRF, you set routing parameters for the VRF that are independent of routing parameters in another VRF for the same routing protocol instance.

You can assign interfaces and route protocols to a VRF to create virtual Layer 3 networks. An interface exists in only one VRF. The following figure shows one physical network split into two virtual networks with two VRFs. Routers Z, A, and B exist in VRF Red and form one address domain. These routers share route updates that do not include Router C because Router C is configured in a different VRF.

***Figure 13: VRFs in a Network***




By default, Cisco NX-OS uses the VRF of the incoming interface to select which routing table to use for a route lookup. You can configure a route policy to modify this behavior and set the VRF that Cisco NX-OS uses for incoming packets.

Cisco NX-OS supports route leaking (import or export) between VRFs.

## Route Leaking and Importing Routes from the Default VRF

Cisco NX-OS supports route leaking (import or export) between VRFs.

You can import IP prefixes from the global routing table (the default VRF) into any other VRF by using an import policy. The VRF import policy uses a route map to specify the prefixes to be imported into a VRF. The policy can import IPv4 unicast prefixes.

**Note** Routes in the BGP default VRF can be imported directly. Any other routes in the default VRF should be redistributed into BGP first.

IP prefixes are defined as match criteria for the import route map through standard route policy filtering mechanisms. For example, you can create an IP prefix list or an as-path filter to define an IP prefix or IP prefix range and use that prefix list or as-path filter in a match clause for the route map. Prefixes that pass through the route map are imported into the specified VRF using the import policy. IP prefixes that are imported into a VRF through this import policy cannot be reimported into another VRF.

For more information, see the *Guidelines and Limitations for VRF Route Leaking* section.

# VRF-Aware Services

A fundamental feature of the Cisco NX-OS architecture is that every IP-based feature is VRF aware.

The following VRF-aware services can select a particular VRF to reach a remote server or to filter information based on the selected VRF:

- AAA—See the *Cisco Nexus Security Configuration Guide* for more information.

- Callhome—See the *Cisco Nexus System Management Configuration Guide* for more information.
- NTP—See the *Cisco Nexus System Management Configuration Guide* for more information.
- RADIUS—See the *Cisco Nexus Security Configuration Guide* for more information.
- SNMP—See the *Cisco Nexus System Management Configuration Guide* for more information.
- SSH—See the *Cisco Nexus Security Configuration Guide* for more information.
- Syslog—See the *Cisco Nexus System Management Configuration Guide* for more information.
- TACACS+—See the *Cisco Nexus Security Configuration Guide* for more information.
- VRRP—See *Configuring VRRP* section for more information.

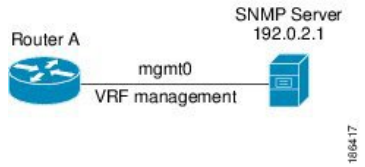
See the appropriate configuration guide for each service for more information on configuring VRF support in that service.

## Reachability

Reachability indicates which VRF contains the routing information necessary to get to the server providing the service. For example, you can configure an SNMP server that is reachable on the management VRF. When you configure that server address on the router, you also configure which VRF Cisco NX-OS must use to reach the server.

The following figure shows an SNMP server that is reachable over the management VRF. You configure Router A to use the management VRF for SNMP server host 192.0.2.1.

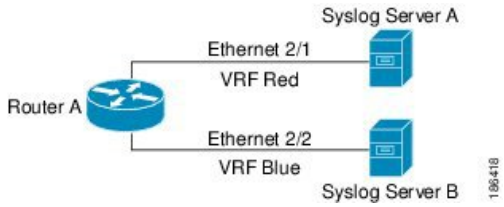
***Figure 14: Service VRF Reachability***




## Filtering

Filtering allows you to limit the type of information that goes to a VRF-aware service based on the VRF. For example, you can configure a syslog server to support a particular VRF. The following figure shows two syslog servers with each server supporting one VRF. Syslog server A is configured in VRF Red, so Cisco NX-OS sends only system messages generated in VRF Red to syslog server A.

***Figure 15: Service VRF Filtering***

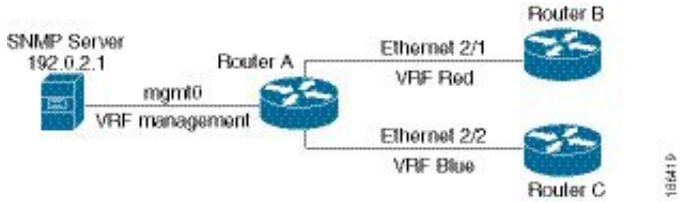


## Combining Reachability and Filtering

You can combine reachability and filtering for VRF-aware services. You can configure the VRF that Cisco NX-OS uses to connect to that service as well as the VRF that the service supports. If you configure a service in the default VRF, you can optionally configure the service to support all VRFs.

The following figure shows an SNMP server that is reachable on the management VRF. You can configure the SNMP server to support only the SNMP notifications from VRF Red, for example.

***Figure 16: Service VRF Reachability Filtering***




# Guidelines and Limitations for VRFs

VRFs have the following configuration guidelines and limitations:

- Names in the prefix-list are case-insensitive. We recommend using unique names. Do not use the same name by modifying upper-case and lower-case characters. For example, CTCPrimaryNetworks and CtcPrimaryNetworks are not two different entries.
- Cisco NX-OS creates the default and management VRFs by default. You should make the mgmt0 interface a member of the management VRF.
- The **write erase boot** command does not remove the management VRF configurations. You must use the **write erase** command and then the **write erase boot** command.
- When you make an interface a member of an existing VRF, Cisco NX-OS removes all Layer 3 configurations. You should configure all Layer 3 parameters after adding an interface to a VRF.
- You should add the mgmt0 interface to the management VRF and configure the mgmt0 IP address and other parameters after you add it to the management VRF.
- If you configure an interface for a VRF before the VRF exists, the interface is operationally down until you create the VRF.

# Guidelines and Limitations for VRF Route Leaking

VRF route leaking has the following configuration guidelines and limitations:

- Route leaking is supported between any two non-default VRFs and from the default VRF to a non-default VRF.

**Note** Route leaking between VRFs is not supported for MPLS Segment Routing (SR-MPLS).

Route leaking between VRFs is not supported for BGP. A BGP speaker cannot connect to a peer IP that is routed through a different VRF.

- Route leaking to the default VRF is not allowed because it is the global VRF.
- You can restrict route leaking to specific routes using route map filters to match designated IP addresses.
- By default, the maximum number of IP prefixes that can be imported from the default VRF into a non-default VRF is 1000 routes.
- There is no limit on the number of routes that can be leaked between two non-default VRFs.

# Default Settings

The table lists the default settings for VRF parameters.

***Table 7: Default VRF Parameters***

| Parameters | Default |
|---|---|
| Configured VRFs | Default, management |
| Routing context | Default VRF |

# Configuring VRFs

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

## Creating a VRF

You can create a VRF.

**Note** Any commands available in global configuration mode are also available in VRF configuration mode.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | switch# **configure terminal** | Enters global configuration mode. |
| **Step 2** | [**no**] **vrf context** *name*<br>**Example:**<br>`switch(config)# vrf context Enterprise`<br>`switch(config-vrf)#` | Creates a new VRF and enters VRF configuration mode. The *name* can be any case-sensitive, alphanumeric string up to 32 characters.<br>Using the **no** option with this command deletes the VRF and all associated configurations. |
| **Step 3** | (Optional) **ip route** {*ip-prefix* \| *ip-addr ip-mask*} {[*next-hop* \| *nh-prefix*] \| [*interface next-hop* \| *nh-prefix*]} [**tag** *tag-value* [*preference*]<br>**Example:**<br>`switch(config-vrf)# ip route 192.0.2.0/8 ethernet 1/2 192.0.2.4` | Configures a static route and the interface for this static route. You can optionally configure the next-hop address. The *preference* value sets the administrative distance. The range is from 1 to 255. The default is 1. |
| **Step 4** | (Optional) **show vrf** [*vrf-name*]<br>**Example:**<br>`switch(config-vrf)# show vrf Enterprise` | Displays VRF information. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-vrf)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example show how to create a VRF and add a static route to the VRF:

```
switch# configure terminal
switch(config)# vrf context Management
switch(config-vrf)# ip route 0.0.0.0/8 ethernet 1/2
switch(config-vrf)# exit
switch(config)# copy running-config startup-config
```

# Assigning VRF Membership to an Interface

You can make an interface a member of a VRF.

### Before you begin

Assign the IP address for an interface after you have configured the interface for a VRF.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/2`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 2** | **vrf member** *vrf-name*<br>**Example:**<br>`switch(config-if)# vrf member RemoteOfficeVRF` | Adds this interface to a VRF. |
| **Step 3** | **ip address** *ip-prefix/length*<br>**Example:**<br>`switch(config-if)# ip address 192.0.2.1/16` | Configures an IP address for this interface. You must do this step after you assign this interface to a VRF. |
| **Step 4** | (Optional) **show vrf** *vrf-name* **interface** *interface-type number*<br>**Example:**<br>`switch(config-vrf)# show vrf Enterprise interface ethernet 1/2` | Displays VRF information. |
| **Step 5** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-vrf)# copy running-config startup-config` | Saves this configuration change. |

**Example**

This example shows how to add an interface to the VRF:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# copy running-config startup-config
```

# Configuring VRF Parameters for a Routing Protocol

You can associate a routing protocol with one or more VRFs. See the appropriate chapter for information on how to configure VRFs for the routing protocol. This section uses OSPFv2 as an example protocol for the detailed configuration steps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **router ospf** *instance-tag*<br>**Example:**<br>`switch (config-vrf)# router ospf 201`<br>`switch(config-router)#` | Creates a new OSFPv2 instance with the configured instance tag. |
| **Step 2** | **vrf** *vrf-name*<br>**Example:**<br>`switch(config-router)# vrf RemoteOfficeVRF`<br>`switch(config-router-vrf)#` | Enters VRF configuration mode. |
| **Step 3** | (Optional) **maximum-paths** *paths*<br>**Example:**<br>`switch(config-router-vrf)# maximum-paths 4` | Configures the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF. This command is used for load balancing. |
| **Step 4** | **exit**<br>**Example:**<br>`switch(config-router-vrf)# exit`<br>`switch(config-router)#` | Exits VRF configuration mode. |
| **Step 5** | **exit**<br>**Example:**<br>`switch(config-router)# exit`<br>`switch(config)#` | Exits router configuration mode. |
| **Step 6** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/2`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 7** | **vrf member** *vrf-name*<br>**Example:**<br>`switch(config-if)# vrf member RemoteOfficeVRF` | Adds this interface to a VRF. |
| **Step 8** | **ip address** *ip-prefix/length*<br>**Example:**<br>`switch(config-if)# ip address 192.0.2.1/16` | Configures an IP address for this interface. You must do this step after you assign this interface to a VRF. |
| **Step 9** | **ip router ospf** *instance-tag* **area** *area-id*<br>**Example:**<br>`switch(config-if)# ip router ospf 201 area 0` | Assigns this interface to the OSPFv2 instance and area configured. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 10** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

### Example

This example shows how to create a VRF and add an interface to the VRF:

```
switch# configure terminal
switch(config)# vrf context RemoteOfficeVRF
switch(config-vrf)# exit
switch(config)# router ospf 201
switch(config-router)# vrf RemoteOfficeVRF
switch(config-router-vrf)# maximum-paths 4
switch(config-router-vrf)# interface ethernet 1/2
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router ospf 201 area 0
switch(config-if)# exit
switch(config)# copy running-config startup-config
```

# Configuring a VRF-Aware Service

You can configure a VRF-aware service for reachability and filtering.

This section uses SNMP and IP domain lists as example services for the detailed configuration steps.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **snmp-server host** *ip-address* [**filter-vrf** *vrf-name*] [**use-vrf** *vrf-name*]<br>**Example:**<br>`switch(config)# snmp-server host 192.0.2.1 use-vrf Red` | Configures a global SNMP server and configures the VRF that Cisco NX-OS uses to reach the service. Use the **filter-vrf** keyword to filter information from the selected VRF to this server. |
| **Step 2** | **vrf context** *vrf-name*<br>**Example:**<br>`switch(config)# vrf context Blue`<br>`switch(config-vrf)#` | Creates a new VRF. |
| **Step 3** | **ip domain-list** *domain-name* [**all-vrfs**] [**use-vrf** *vrf-name*]<br>**Example:**<br>`switch(config-vrf)# ip domain-list List all-vrfs use-vrf Blue` | Configures the domain list in the VRF and optionally configures the VRF that Cisco NX-OS uses to reach the domain name listed. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-vrf)# copy running-config startup-config` | Saves this configuration change. |

### Example

This example shows how to send SNMP information for all VRFs to SNMP host 192.0.2.1, reachable on VRF Red:

```
switch# configure terminal
switch(config)# snmp-server host 192.0.2.1 for-all-vrfs use-vrf Red
switch(config)# copy running-config startup-config
```

This example shows how to filter SNMP information for VRF Blue to SNMP host 192.0.2.12, reachable on VRF Red:

```
switch# configure terminal
switch(config)# vrf context Blue
switch(config-vrf)# snmp-server host 192.0.2.12 use-vrf Red
switch(config)# copy running-config startup-config
```

# Setting the VRF Scope

You can set the VRF scope for all EXEC commands (for example, **show** commands). Doing so automatically restricts the scope of the output of EXEC commands to the configured VRF. You can override this scope by using the VRF keywords available for some EXEC commands.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **routing-context vrf** *vrf-name*<br>**Example:**<br>`switch# routing-context vrf red`<br>`switch%red#` | Sets the routing context for all EXEC commands. The default routing context is the default VRF.<br>**Note** Use the **routing-context vrf default** command to return to the default VRF scope. |

### Example

To return to the default VRF scope, use the following command in EXEC mode:

| Command | Purpose |
| --- | --- |
| **routing-context vrf default**<br>Example:<br>`switch%red# routing-context vrf default`<br>`switch#` | Sets the default routing context. |

# Verifying the VRF Configuration

To display VRF configuration information, perform one of the following tasks:

| Command | Purpose |
| --- | --- |
| **show bgp process vrf** [ *vrf-name* ] | Displays the information for all or one VRF. |
| **show vrf** [*vrf-name*] | Displays the information for all or one VRF. |
| **show vrf** [*vrf-name*] **detail** | Displays detailed information for all or one VRF. |
| **show vrf** [*vrf-name*] [**interface** *interface-type slot/port*] | Displays the VRF status for an interface. |

# Configuration Examples for VRFs

This example shows how to configure VRF Red, add an SNMP server to that VRF, and add an instance of OSPF to VRF Red:

```
vrf context Red
    snmp-server host 192.0.2.12 use-vrf Red
    router ospf 201

vrf Red
    interface ethernet 1/2
    vrf member Red
    ip address 192.0.2.1/16
    ip router ospf 201 area 0
```

This example shows how to configure VRF Red and Blue, add an instance of OSPF to each VRF, and create an SNMP context for each OSPF instance in each VRF:

```
vrf context Red
vrf context Blue
vrf context Green

feature ospf
    router ospf Lab
    vrf Red

router ospf Production
    vrf Blue
```

```
    router-id 1.1.1.1
    vrf Green
    router-id 2.2.2.2

interface ethernet 1/2
    vrf member Red
    ip address 192.0.2.1/16
    ip router ospf Lab area 0
    no shutdown

interface ethernet 10/2
    vrf member Blue
    ip address 192.0.2.1/16
    ip router ospf Production area 0
    no shutdown

interface ethernet 10/3
    vrf member Green
    ip address 192.0.2.1/16
    ip router ospf Production area 0
    no shutdown

snmp-server user admin network-admin auth md5 nbv-12345
    snmp-server community public ro

snmp-server context lab instance Lab vrf Red
    snmp-server context production instance Production vrf Blue
```

Use the SNMP context **lab** to access the OSPF-MIB values for the OSPF instance Lab in VRF Red in this example.

This example shows how to configure route leaking between two non-default VRFs and from the default VRF to a non-default VRF:

```
feature bgp
vrf context Green
    ip route 33.33.33.33/32 35.35.1.254
    address-family ipv4 unicast
    route-target import 3:3
    route-target export 2:2
    export map test
    import map test
    import vrf default map test

interface Ethernet1/7
    vrf member Green
    ip address 35.35.1.2/24

vrf context Shared
    ip route 44.44.44.44/32 45.45.1.254
    address-family ipv4 unicast
    route-target import 1:1
    route-target import 2:2
    route-target export 3:3
    export map test
    import map test
    import vrf default map test

interface Ethernet1/11
    vrf member Shared
    ip address 45.45.1.2/24

router bgp 100
```

```
    address-family ipv4 unicast
    redistribute static route-map test
    vrf Green
    address-family ipv4 unicast
    redistribute static route-map test
    vrf Shared
    address-family ipv4 unicast
    redistribute static route-map test

ip prefix-list test seq 5 permit 0.0.0.0/0 le 32
    route-map test permit 10
    match ip address prefix-list test

ip route 100.100.100.100/32 55.55.55.1
    switch# show ip route vrf all
    IP Route Table for VRF "default"
    '*' denotes best ucast next-hop
    '**' denotes best mcast next-hop
    '[x/y]' denotes [preference/metric]
    '%<string>' in via output denotes VRF <string>

55.55.55.0/24, ubest/mbest: 1/0, attached
    *via 55.55.55.5, Lo0, [0/0], 00:07:59, direct
    55.55.55.5/32, ubest/mbest: 1/0, attached
    *via 55.55.55.5, Lo0, [0/0], 00:07:59, local
    100.100.100.100/32, ubest/mbest: 1/0
    *via 55.55.55.1, [1/0], 00:07:42, static

IP Route Table for VRF "management"
    '*' denotes best ucast next-hop
    '**' denotes best mcast next-hop
    '[x/y]' denotes [preference/metric]
    '%<string>' in via output denotes VRF <string>

0.0.0.0/0, ubest/mbest: 1/0
    *via 10.29.176.1, [1/0], 12:53:54, static
    10.29.176.0/24, ubest/mbest: 1/0, attached
    *via 10.29.176.233, mgmt0, [0/0], 13:11:57, direct
    10.29.176.233/32, ubest/mbest: 1/0, attached
    *via 10.29.176.233, mgmt0, [0/0], 13:11:57, local

IP Route Table for VRF "Green"
    '*' denotes best ucast next-hop
    '**' denotes best mcast next-hop
    '[x/y]' denotes [preference/metric]
    '%<string>' in via output denotes VRF <string>
    33.33.33.33/32, ubest/mbest: 1/0
    *via 35.35.1.254, [1/0], 00:23:44, static
    35.35.1.0/24, ubest/mbest: 1/0, attached
    *via 35.35.1.2, Eth1/7, [0/0], 00:26:46, direct
    35.35.1.2/32, ubest/mbest: 1/0, attached
    *via 35.35.1.2, Eth1/7, [0/0], 00:26:46, local
    44.44.44.44/32, ubest/mbest: 1/0
    *via 45.45.1.254%Shared, [20/0], 00:12:08, bgp-100, external, tag 100
    100.100.100.100/32, ubest/mbest: 1/0
    *via 55.55.55.1%default, [20/0], 00:07:41, bgp-100, external, tag 100

IP Route Table for VRF "Shared"
    '*' denotes best ucast next-hop
    '**' denotes best mcast next-hop
    '[x/y]' denotes [preference/metric]
    '%<string>' in via output denotes VRF <string>

33.33.33.33/32, ubest/mbest: 1/0
```

```
    *via 35.35.1.254%Green, [20/0], 00:12:34, bgp-100, external, tag 100
    44.44.44.44/32, ubest/mbest: 1/0
    *via 45.45.1.254, [1/0], 00:23:16, static
    45.45.1.0/24, ubest/mbest: 1/0, attached
    *via 45.45.1.2, Eth1/11, [0/0], 00:25:53, direct
    45.45.1.2/32, ubest/mbest: 1/0, attached
    *via 45.45.1.2, Eth1/11, [0/0], 00:25:53, local
    100.100.100.100/32, ubest/mbest: 1/0
    *via 55.55.55.1%default, [20/0], 00:07:41, bgp-100, external, tag 100
switch(config)#
```

The following example shows how to allow re-importation of already imported routes that is introduced in the “export vrf default” command to allow VPN imported routes to be re-imported into the default-VRF.

```
vrf context vpn1
    address-family ipv4 unicast
        export vrf default [<prefix-limit>] map <route-map> [allow-vpn]
    address-family ipv6 unicast
        export vrf default [<prefix-limit>] map <route-map> [allow-vpn]
```

The following example shows BGP IPv4 Unicast configuration.

```
bl1(config-vrf)# show bgp ipv4 unicast 11.11.11.11/32
BGP routing table information for VRF default, address family IPv4 Unicast
BGP routing table entry for 11.11.11.11/32, version 14
Paths: (1 available, best #1)
Flags: (0x08041a) on xmit-list, is in urib, is best urib route, is in HW

  Advertised path-id 1
  Path type: internal, path is valid, is best path, in rib
             Imported from 3.3.3.3:3:11.11.11.11/32 (VRF vni100)
  AS-Path: 150 , path sourced external to AS
    1.1.1.1 (metric 81) from 101.101.101.101 (101.101.101.101)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 100
      Extcommunity:
          RT:100:100
          ENCAP:8
          Router MAC:5254.004e.a437
      Originator: 1.1.1.1 Cluster list: 101.101.101.101

  Path-id 1 advertised to peers:
    30.0.0.2

bl1(config-vrf)# show bgp vrf vni100 ipv4 unicast 11.11.11.11/32
BGP routing table information for VRF vni100, address family IPv4 Unicast
BGP routing table entry for 11.11.11.11/32, version 8
Paths: (1 available, best #1)
Flags: (0x08041e) on xmit-list, is in urib, is best urib route, is in HW
  vpn: version 19, (0x100002) on xmit-list

  Advertised path-id 1, VPN AF advertised path-id 1
  Path type: internal, path is valid, is best path, in rib
             Imported from 1.1.1.1:3:[5]:[0]:[0]:[32]:[11.11.11.11]:[0.0.0.0]/224
  AS-Path: 150 , path sourced external to AS
    1.1.1.1 (metric 81) from 101.101.101.101 (101.101.101.101)
      Origin incomplete, MED 0, localpref 100, weight 0
      Received label 100
      Extcommunity:
          RT:100:100
          ENCAP:8
          Router MAC:5254.004e.a437
      Originator: 1.1.1.1 Cluster list: 101.101.101.101
```

```
  VRF advertise information:
  Path-id 1 not advertised to any peer

  VPN AF advertise information:
  Path-id 1 not advertised to any peer
```

The following example shows the output of show ipv4 route command

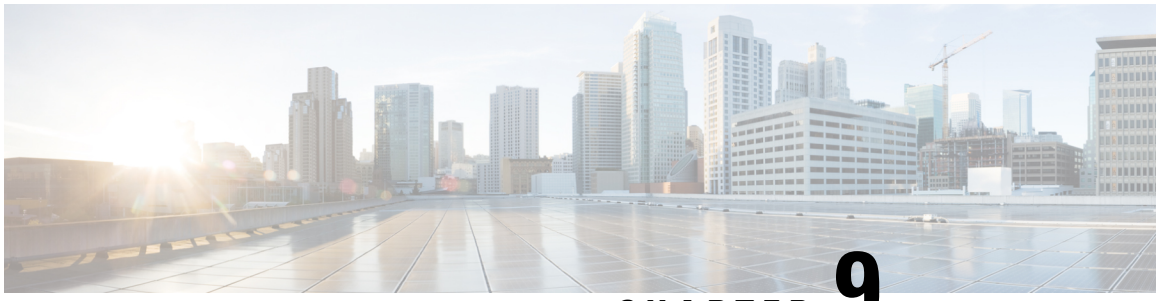
```
bl1(config-if)# show ip route
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

0.0.0.0/0, ubest/mbest: 1/0
    *via vrf vni100, Null0, [20/0], 1d04h, bgp-100, external, tag 100
1.1.1.1/32, ubest/mbest: 1/0
    *via 103.0.0.1, Eth1/1, [110/81], 1d04h, ospf-100, intra
2.2.2.2/32, ubest/mbest: 1/0
    *via 103.0.0.1, Eth1/1, [110/81], 1d04h, ospf-100, intra
3.3.3.3/32, ubest/mbest: 2/0, attached
    *via 3.3.3.3, Lo0, [0/0], 1d04h, local
    *via 3.3.3.3, Lo0, [0/0], 1d04h, direct
9.9.9.9/32, ubest/mbest: 1/0, attached
    *via 9.9.9.9%vni100, Lo9, [20/0], 1d03h, bgp-100, external, tag 100
30.0.0.0/24, ubest/mbest: 1/0, attached
    *via 30.0.0.1, Eth1/2, [0/0], 1d04h, direct
30.0.0.1/32, ubest/mbest: 1/0, attached
    *via 30.0.0.1, Eth1/2, [0/0], 1d04h, local
33.33.33.33/32, ubest/mbest: 1/0
    *via 30.0.0.2, [20/0], 1d04h, bgp-100, external, tag 300
100.0.0.0/24, ubest/mbest: 1/0, attached
    *via 100.0.0.3%vni100, Vlan100, [20/0], 1d04h, bgp-100, external, tag 100
101.0.0.0/24, ubest/mbest: 1/0
    *via 103.0.0.1, Eth1/1, [110/80], 1d04h, ospf-100, intra
101.101.101.101/32, ubest/mbest: 1/0
    *via 103.0.0.1, Eth1/1, [110/41], 1d04h, ospf-100, intra
102.0.0.0/24, ubest/mbest: 1/0
    *via 103.0.0.1, Eth1/1, [110/80], 1d04h, ospf-100, intra
103.0.0.0/24, ubest/mbest: 1/0, attached
    *via 103.0.0.2, Eth1/1, [0/0], 1d04h, direct
103.0.0.2/32, ubest/mbest: 1/0, attached
```

# Additional References

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

## Related Documents for VRFs

| Related Topic | Document Title |
|---|---|
| VRFs | *Cisco Nexus® 3550-T System Management Configuration* section |



CHAPTER 9

# Configuring VRRP

This chapter contains the following sections:



# About VRRP

VRRP allows for a transparent failover at the first-hop IP router by configuring a group of routers to share a virtual IP address. VRRP selects an allowed router in that group to handle all packets for the virtual IP address. The remaining routers are in standby and take over if the allowed router fails.

## VRRP Operation

A LAN client can determine which router should be the first hop to a particular remote destination by using a dynamic process or static configuration. Examples of dynamic router discovery are as follows:

Proxy ARP—The client uses Address Resolution Protocol (ARP) to get the destination it wants to reach, and a router responds to the ARP request with its own MAC address.

Routing protocol—The client listens to dynamic routing protocol updates.

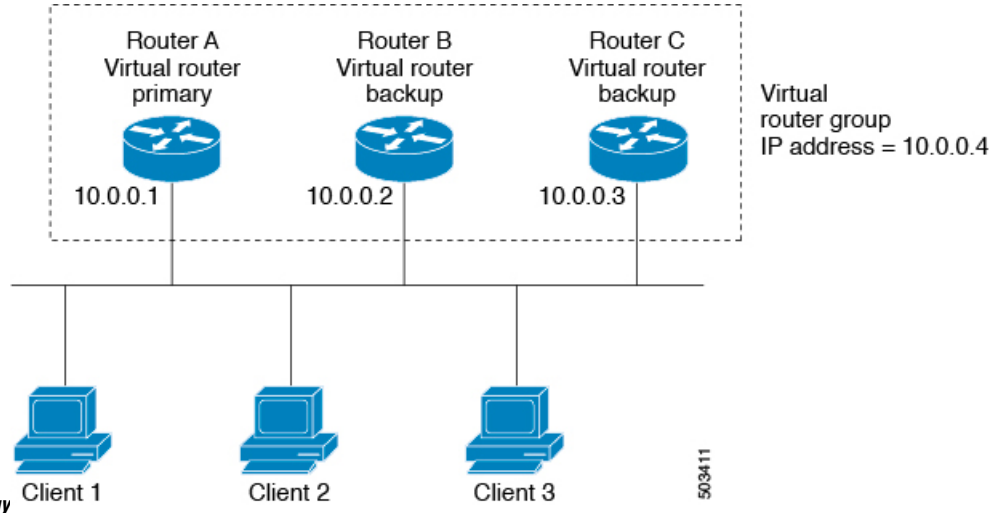
The disadvantage to dynamic discovery protocols is that they incur some configuration and processing overhead on the LAN client. Also, if a router fails, the process of switching to another router can be slow.

An alternative to dynamic discovery protocols is to statically configure a default router on the client. Although this approach simplifies client configuration and processing, it creates a single point of failure. If the default gateway fails, the LAN client is limited to communicating only on the local IP network segment and is cut off from the rest of the network.

VRRP can solve the static configuration problem by enabling a group of routers (a VRRP group) to share a single virtual IP address. You can then configure the LAN clients with the virtual IP address as their default gateway.

The following figure shows a basic VLAN topology. In this example, Routers A, B, and C form a VRRP group. The IP address of the group is the same address that was configured for the Ethernet interface of Router A (10.0.0.1).

*Figure 17: Basic VRRP*




*Topology*

Because the virtual IP address uses the IP address of the physical Ethernet interface of Router A, Router A is the primary (also known as the IP address owner). As the primary, Router A owns the virtual IP address of the VRRP group and forwards packets sent to this IP address. Clients 1 through 3 are configured with the default gateway IP address of 10.0.0.1.

Routers B and C function as backups. If the primary fails, the backup router with the highest priority becomes the primary and takes over the virtual IP address to provide uninterrupted service for the LAN hosts. When Router A recovers, it becomes the primary again.

**Note** Packets received on a routed port destined for the VRRP virtual IP address terminate on the local router, regardless of whether that router is the primary VRRP router or a backup VRRP router. These packets include ping and Telnet traffic. Packets received on a Layer 2 (VLAN) interface destined for the VRRP virtual IP address terminate on the primary router.

# VRRP Benefits

The benefits of VRRP are as follows:

- Redundancy—Enables you to configure multiple routers as the default gateway router, which reduces the possibility of a single point of failure in a network.
- Load sharing—Allows traffic to and from LAN clients to be shared by multiple routers. The traffic load is shared more equitably among available routers.

- Multiple VRRP groups—Supports multiple VRRP groups on a router physical interface if the platform supports multiple MAC addresses. Multiple VRRP groups enable you to implement redundancy and load sharing in your LAN topology.
- Multiple IP addresses—Allows you to manage multiple IP addresses, including secondary IP addresses. If you have multiple subnets that are configured on an Ethernet interface, you can configure VRRP on each subnet.
- Preemption—Enables you to preempt a backup router that has taken over for a failing primary with a higher priority backup router that has become available.
- Advertisement protocol—Uses a dedicated Internet Assigned Numbers Authority (IANA) standard multicast address (224.0.0.18) for VRRP advertisements. This addressing scheme minimizes the number of routers that must service the multicasts and allows test equipment to accurately identify VRRP packets on a segment. IANA has assigned the IP protocol number 112 to VRRP.
- VRRP tracking—Ensures that the best VRRP router is the primary for the group by altering VRRP priorities based on interface states.

# Multiple VRRP Groups

You can configure multiple VRRP groups on a physical interface. For the number of supported VRRP groups, see the *Cisco Nexus® 3550-T Verified Scalability Guide*.

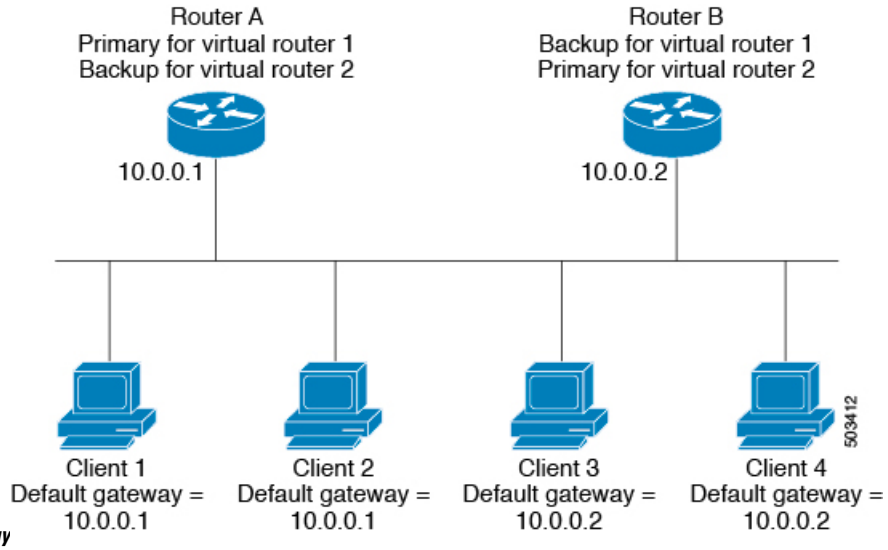
The number of VRRP groups that a router interface can support depends on the following factors:

- Router processing capability
- Router memory capability

In a topology where multiple VRRP groups are configured on a router interface, the interface can act as a primary for one VRRP group and as a backup for one or more other VRRP groups.

The following image shows a LAN topology in which VRRP is configured so that Routers A and B share the traffic to and from clients 1 through 4. Routers A and B act as backups to each other if either router fails.

*Figure 18: Load Sharing and Redundancy VRRP*




*Topology*

This topology contains two virtual IP addresses for two VRRP groups that overlap. For VRRP group 1, Router A is the owner of IP address 10.0.0.1 and is the primary. Router B is the backup to Router A. Clients 1 and 2 are configured with the default gateway IP address of 10.0.0.1.

For VRRP group 2, Router B is the owner of IP address 10.0.0.2 and is the primary. Router A is the backup to router B. Clients 3 and 4 are configured with the default gateway IP address of 10.0.0.2.

# VRRP Router Priority and Preemption

An important aspect of the VRRP redundancy scheme is the VRRP router priority because the priority determines the role that each VRRP router plays and what happens if the primary router fails.

If a VRRP router owns the virtual IP address and the IP address of the physical interface, this router functions as the primary. The priority of the primary is 255.

The priority also determines if a VRRP router functions as a backup router and the order of ascendancy to becoming a primary if the primary fails.

For example, if Router A, the primary in a LAN topology, fails, VRRP must determine if backups B or C should take over. If you configure Router B with priority 101 and Router C with the default priority of 100, VRRP selects Router B to become the primary because it has the higher priority. If you configure Routers B and C with the default priority of 100, VRRP selects the backup with the higher IP address to become the primary.

VRRP uses preemption to determine what happens after a VRRP backup router becomes the primary. With preemption enabled by default, VRRP switches to a backup if that backup comes online with a priority higher than the new primary. For example, if Router A is the primary and fails, VRRP selects Router B (next in order of priority). If Router C comes online with a higher priority than Router B, VRRP selects Router C as the new primary, even though Router B has not failed.

If you disable preemption, VRRP switches only if the original primary recovers or the new primary fails.

# VRRP Advertisements

The VRRP primary sends VRRP advertisements to other VRRP routers in the same group. The advertisements communicate the priority and state of the primary. Cisco NX-OS encapsulates the VRRP advertisements in IP packets and sends them to the IP multicast address assigned to the VRRP group. Cisco NX-OS sends the advertisements once every second by default, but you can configure a different advertisement interval.

# VRRP Authentication

VRRP supports the following authentication functions:

- No authentication
- Plain text authentication

VRRP rejects packets in any of the following cases:

- The authentication schemes differ on the router and in the incoming packet.
- Text authentication strings differ on the router and in the incoming packet.

# VRRP Tracking

VRRP supports the following options for tracking:

- Native interface tracking—Tracks the state of an interface and uses that state to determine the priority of the VRRP router in a VRRP group. The tracked state is down if the interface is down or if the interface does not have a primary IP address.
- Object tracking—Tracks the state of a configured object and uses that state to determine the priority of the VRRP router in a VRRP group. See the *Configuring Object Tracking* section, for more information on object tracking.

If the tracked state (interface or object) goes down, VRRP updates the priority based on what you configure the new priority to be for the tracked state. When the tracked state comes up, VRRP restores the original priority for the virtual router group.

For example, you might want to lower the priority of a VRRP group member if its uplink to the network goes down so another group member can take over as primary for the VRRP group. See the Configuring VRRP Interface State Tracking, on page 199 section for more information.

**Note** VRRP does not support Layer 2 interface tracking.

# High Availability

VRRP supports high availability through stateful restarts and stateful switchovers. A stateful restart occurs when the VRRP process fails and is restarted. A stateful switchover occurs when the active supervisor switches to the standby supervisor. Cisco NX-OS applies the run-time configuration after the switchover.

# Virtualization Support

VRRP supports virtual routing and forwarding (VRF) instances.

# Guidelines and Limitations for VRRP

VRRP has the following configuration guidelines and limitations:

- You cannot configure VRRP on the management interface.
- When VRRP is enabled, you should replicate the VRRP configuration across devices in your network.
- We recommend that you do not configure more than one first-hop redundancy protocol on the same interface.
- You must configure an IP address for the interface on which you configure VRRP and enable that interface before VRRP becomes active.
- Cisco NX-OS removes all Layer 3 configurations on an interface when you change the interface VRF membership or the port channel membership or when you change the port mode to Layer 2.
- When you configure VRRP to track a Layer 2 interface, you must shut down the Layer 2 interface and reenable the interface to update the VRRP priority to reflect the state of the Layer 2 interface.

# Default Settings for VRRP Parameters

The following table lists the default settings for VRRP parameters.

***Table 8: Default VRRP Parameters***

| Parameters | Default |
|---|---|
| VRRP | Disabled |
| Advertisement interval | 1 second |
| Authentication | No authentication |
| Preemption | Enabled |
| Priority | 100 |

# Configuring VRRP

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

# Enabling VRRP

You must globally enable VRRP before you configure and enable any VRRP groups.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | [**no**] **feature vrrp**<br>**Example:**<br>`switch(config)# feature vrrp` | Enables VRRP. Use the **no** form of this command to disable VRRP. |
| **Step 3** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring VRRP Groups

You can create a VRRP group, assign the virtual IP address, and enable the group.

You can configure one virtual IPv4 address for a VRRP group. By default, the primary VRRP router drops the packets addressed directly to the virtual IP address because the VRRP primary is intended only as a next-hop router to forward packets. Some applications require that Cisco NX-OS accept packets that are addressed to the virtual router IP address. Use the secondary option to the virtual IP address to accept these packets when the local router is the VRRP primary.

Once you have configured the VRRP group, you must explicitly enable the group before it becomes active.

### Before you begin

Ensure that you have configured an IP address on the interface. See Configuring IPv4 Addressing, on page 20.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:** | Enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config)# interface ethernet 1/1`<br>`switch(config-if)#` | |
| **Step 3** | **vrrp** *number*<br>**Example:**<br>`switch(config-if)# vrrp 250`<br>`switch(config-if-vrrp)#` | Creates a virtual router group. The range is 1–255. |
| **Step 4** | **address** *ip-address* [**secondary**]<br>**Example:**<br>`switch(config-if-vrrp)# address 192.0.2.8` | Configures the virtual IPv4 address for the specified VRRP group. This address should be in the same subnet as the IPv4 address of the interface.<br>Use the **secondary** option only if applications require that VRRP routers accept the packets sent to the virtual router's IP address and deliver to applications. |
| **Step 5** | **no shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# no shutdown` | Enables the VRRP group, which is disabled by default. |
| **Step 6** | (Optional) **show vrrp**<br>**Example:**<br>`switch(config-if-vrrp)# show vrrp` | Displays a summary of VRRP information. |
| **Step 7** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if-vrrp)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring VRRP Priority

The valid priority range for a virtual router is from 1 to 254 (1 is the lowest priority and 254 is the highest). The default priority value for backups is 100. For devices whose interface IP address is the same as the primary virtual IP address (the primary), the default value is 255.

### Before you begin

Ensure that you have configured an IP address on the interface. See Configuring IPv4 Addressing, on page 20.

Ensure that you have enabled VRRP. (see the Configuring VRRP, on page 192 section).

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/1`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **vrrp** *number*<br>**Example:**<br>`switch(config-if)# vrrp 250`<br>`switch(config-if-vrrp)#` | Creates a virtual router group. |
| **Step 4** | **shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# shutdown` | Disables the VRRP group. |
| **Step 5** | **priority** *level* [**forwarding-threshold lower** *lower-value* **upper** *upper-value*]<br>**Example:**<br>`switch(config-if-vrrp)# priority 60 forwarding-threshold lower 40 upper 50` | Sets the priority level used to select the active router in a VRRP group. The *level* range is 1–254. The default is 100 for backups and 255 for a primary that has an interface IP address equal to the virtual IP address. |
| **Step 6** | **no shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# no shutdown` | Enables the VRRP group. |
| **Step 7** | (Optional) **show vrrp**<br>**Example:**<br>`switch(config-if-vrrp)# show vrrp` | Displays a summary of VRRP information. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if-vrrp)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring VRRP Authentication

You can configure simple text authentication for a VRRP group.

### Before you begin

Ensure that you have configured an IP address on the interface (see Configuring IPv4 Addressing, on page 20).

Ensure that you have enabled VRRP (see the Configuring VRRP, on page 192 section).

Ensure that the authentication configuration is identical for all VRRP devices in the network.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/1`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **vrrp** *number*<br>**Example:**<br>`switch(config-if)# vrrp 250`<br>`switch(config-if-vrrp)#` | Creates a virtual router group. |
| **Step 4** | **shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# shutdown` | Disables the VRRP group. |
| **Step 5** | **authentication text** *password*<br>**Example:**<br>`switch(config-if-vrrp)# authentication text aPassword` | Assigns the simple text authentication option and specifies the keyname password. The keyname range is from 1 to 255 characters. We recommend that you use at least 16 characters. The text password is up to eight alphanumeric characters. |
| **Step 6** | **no shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# no shutdown` | Enables the VRRP group, which is disabled by default. |
| **Step 7** | (Optional) **show vrrp**<br>**Example:**<br>`switch(config-if-vrrp)# show vrrp` | Displays a summary of VRRP information. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if-vrrp)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring Time Intervals for Advertisement Packets

You can configure the time intervals for advertisement packets.

### Before you begin

Ensure that you have configured an IP address on the interface (see Configuring IPv4 Addressing, on page 20).

Ensure that you have enabled VRRP (see the Configuring VRRP, on page 192 section).

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/1`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **vrrp** *number*<br>**Example:**<br>`switch(config-if)# vrrp 250`<br>`switch(config-if-vrrp)#` | Creates a virtual router group. |
| **Step 4** | **shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# shutdown` | Disables the VRRP group. |
| **Step 5** | **advertisement interval** *seconds*<br>**Example:**<br>`switch(config-if-vrrp)#`<br>`advertisement-interval 15` | Sets the interval time in seconds between sending advertisement frames. The range is from 1 to 255. The default is 1 second. |
| **Step 6** | **no shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# no shutdown` | Enables the VRRP group. |
| **Step 7** | (Optional) **show vrrp**<br>**Example:**<br>`switch(config-if-vrrp)# show vrrp` | Displays a summary of VRRP information. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:** | Copies the running configuration to the startup configuration. |

| | Command or Action | Purpose |
|---|---|---|
| | `switch(config-if-vrrp)# copy running-config startup-config` | |

# Disabling Preemption

You can disable preemption for a VRRP group member. If you disable preemption, a higher-priority backup router does not take over for a lower-priority primary router. Preemption is enabled by default.

### Before you begin

Ensure that you have configured an IP address on the interface. See Configuring IPv4 Addressing, on page 20.

Ensure that you have enabled VRRP. See the Configuring VRRP, on page 192 section.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/1`<br>`switch(config-if)#` | Enters interface configuration mode. |
| **Step 3** | **vrrp** *number*<br>**Example:**<br>`switch(config-if)# vrrp 250`<br>`switch(config-if-vrrp)#` | Creates a virtual router group. |
| **Step 4** | **shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# shutdown` | Disables the VRRP group. |
| **Step 5** | **no preempt**<br>**Example:**<br>`switch(config-if-vrrp)# no preempt` | Disables the preempt option and allows the primary to remain when a higher-priority backup appears. |
| **Step 6** | **no shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# no shutdown` | Enables the VRRP group. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 7** | (Optional) **show vrrp**<br>**Example:**<br>`switch(config-if-vrrp)# show vrrp` | Displays a summary of VRRP information. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if-vrrp)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring VRRP Interface State Tracking

Interface state tracking changes the priority of the virtual router based on the state of another interface in the device. When the tracked interface goes down or the IP address is removed, Cisco NX-OS assigns the tracking priority value to the virtual router. When the tracked interface comes up and an IP address is configured on this interface, Cisco NX-OS restores the configured priority to the virtual router (see the Configuring VRRP Priority, on page 194 section).

**Note** VRRP does not support Layer 2 interface tracking.

### Before you begin

Ensure that you have configured an IP address on the interface (see Configuring IPv4 Addressing, on page 20).

Ensure that you have enabled VRRP (see the Configuring VRRP, on page 192 section).

Ensure that you have enabled the virtual router (see the Configuring VRRP Groups, on page 193 section).

Ensure that you have enabled preemption on the interface.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |
| **Step 2** | **interface** *interface-type slot/port*<br>**Example:**<br>`switch(config)# interface ethernet 1/1`<br>`switch(config-if)#` | Enters interface configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **vrrp** *number*<br>**Example:**<br>`switch(config-if)# vrrp 250`<br>`switch(config-if-vrrp)#` | Creates a virtual router group. |
| **Step 4** | **shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# shutdown` | Disables the VRRP group. |
| **Step 5** | **track interface** *type slot/port* **priority** *value*<br>**Example:**<br>`switch(config-if-vrrp)# track interface ethernet 1/10 priority 254` | Enables interface priority tracking for a VRRP group. The priority range is from 1 to 254. |
| **Step 6** | **no shutdown**<br>**Example:**<br>`switch(config-if-vrrp)# no shutdown` | Enables the VRRP group. |
| **Step 7** | (Optional) **show vrrp**<br>**Example:**<br>`switch(config-if-vrrp)# show vrrp` | Displays a summary of VRRP information. |
| **Step 8** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if-vrrp)# copy running-config startup-config` | Copies the running configuration to the startup configuration. |

# Configuring VRRP Object Tracking

You can track an IPv4 object using VRRP.

### Before you begin

Make sure that VRRP is enabled.

Configure object tracking using the commands in the *Configuring Object Tracking* section.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure terminal**<br>**Example:**<br>`switch# configure terminal`<br>`switch(config)#` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **interface type number**<br>**Example:**<br>`switch(config)#`<br>`switch(config-if)# interface ethernet 1/1`<br>`switch(config-if)#` | Specifies an interface and enters interface configuration mode. |
| **Step 3** | **vrrp** *number* **address-family ipv4**<br>**Example:**<br>`switch(config-if)# vrrp 5 address-family ipv4`<br>`switch(config-if-vrrp-group)#` | Creates a VRRP group for IPv4 and enters VRRP vrrp number address-family ipv4 group configuration mode. The range is from 1 to 255. |
| **Step 4** | **track** *object-number* **decrement** *number*<br>**Example:**<br>`switch(config-if-vrrp-group)# track 1 decrement 2` | Creates a virtual router group. The range is from 1 to 255. |
| **Step 5** | (Optional) **show running-config vrrp**<br>**Example:**<br>`switch(config-if-vrrp-group)# show running-config vrrp` | Displays the running configuration for VRRP. |
| **Step 6** | (Optional) **copy running-config startup-config**<br>**Example:**<br>`switch(config-if-vrrp-group)# copy running-config startup-config` | Saves this configuration change. |

# Verifying the VRRP Configuration

To display VRRP configuration information, perform one of the following tasks:

| Command | Purpose |
|---|---|
| **show interface** *interface-type* | Displays the virtual router configuration for an interface. |
| **show fhrp** *interface-type interface-number* | Displays First Hop Redundancy Protocol (FHRP) information. |
| **show vrrp** [*group-number*] | Displays the VRRP status for all groups or for a specific VRRP group. |

# Monitoring and Clearing VRRP Statistics

To display VRRP statistics, use the following commands:

| Command | Purpose |
|---|---|
| **show vrrp statistics** | Displays the VRRP statistics. |

Use the **clear vrrp statistics** command to clear the VRRP statistics for all interfaces on the device.

# Configuration Examples for VRRP

In this example, Router A and Router B each belong to three VRRP groups. In the configuration, each group has the following properties:

- Group 1:
  - Virtual IP address is 10.1.0.10.
  - Router A becomes the primary for this group with priority 120.
  - Advertising interval is 3 seconds.
  - Preemption is enabled.
- Group 5:
  - Router B becomes the primary for this group with priority 200.
  - Advertising interval is 30 seconds.
  - Preemption is enabled.
- Group 100:
  - Router A becomes the primary for this group first because it has a higher IP address (10.1.0.2).
  - Advertising interval is the default of 1 second.
  - Preemption is disabled.

Router A

```
switch (config)# interface ethernet 1/1
switch (config-if)# ip address 10.1.0.2/16
switch (config-if)# no shutdown
switch (config-if)# vrrp 1
switch (config-if-vrrp)# priority 120
switch (config-if-vrrp)# authentication text cisco
switch (config-if-vrrp)# advertisement-interval 3
switch (config-if-vrrp)# address 10.1.0.10
switch (config-if-vrrp)# no shutdown
switch (config-if-vrrp)# exit
switch (config-if)# vrrp 5
switch (config-if-vrrp)# priority 100
switch (config-if-vrrp)# advertisement-interval 30
switch (config-if-vrrp)# address 10.1.0.50
switch (config-if-vrrp)# no shutdown
switch (config-if-vrrp)# exit
switch (config-if)# vrrp 100
switch (config-if-vrrp)# no preempt
```

```
switch (config-if-vrrp)# address 10.1.0.100
switch (config-if-vrrp)# no shutdown
```

Router B

```
switch (config)# interface ethernet 1/1
switch (config-if)# ip address 10.2.0.1/2
switch (config-if)# no shutdown
switch (config-if)# vrrp 1
switch (config-if-vrrp)# priority 100
switch (config-if-vrrp)# authentication text cisco
switch (config-if-vrrp)# advertisement-interval 3
switch (config-if-vrrp)# address 10.2.0.10
switch (config-if-vrrp)# no shutdown
switch (config-if-vrrp)# exit
switch (config-if)# vrrp 5
switch (config-if-vrrp)# priority 200
switch (config-if-vrrp)# advertisement-interval 30
switch (config-if-vrrp)# address 10.2.0.50
switch (config-if-vrrp)# no shutdown
switch (config-if-vrrp)# exit
switch (config-if)# vrrp 100
switch (config-if-vrrp)# no preempt
switch (config-if-vrrp)# address 10.2.0.100
switch (config-if-vrrp)# no shutdown
```