Preface

This document describes the configuration details for Cisco NX-OS unicast routing in Cisco Nexus 6000 switches.

This chapter includes the following sections:

- Audience, page 1
- Organization, page 1
- Document Conventions, page 2
- Related Documentation, page 3
- Obtaining Documentation and Submitting a Service Request, page 4

Audience

To use this guide, you must be familiar with IP and routing technology.

Organization

This document is organized into the following chapters:

Title	Description
Chapter 1, “Overview”	Presents an overview of unicast routing and brief descriptions of each feature.
Chapter 1, “Configuring IPv4”	Describes how to configure and manage IPv4, including ARP and ICMP.
Chapter 1, “Configuring IPv6”	Describes how to configure and manage IPv6, including ARP and ICMP.
Chapter 1, “Configuring OSPFv2”	Describes how to configure the OSPFv2 routing protocol for IPv4 networks.
Chapter 1, “Configuring OSPFv3”	Describes how to configure the OSPFv3 routing protocol for IPv6 networks.
Chapter 1, “Configuring EIGRP”	Describes how to configure the Cisco EIGRP routing protocol for IPv4 networks.

Title	Description
Chapter 1, “Configuring Basic BGP”	Describes how to configure basic features for the BGP routing protocol for IPv4 networks.
Chapter 1, “Configuring Advanced BGP”	Describes how to configure advanced features for the BGP routing protocol for IPv4 networks, including route redistribution and route aggregation.
Chapter 1, “Configuring RIP”	Describes how to configure the RIP routing protocols for IPv4 networks.
Chapter 1, “Configuring Static Routing”	Describes how to configure static routing for IPv4 networks.
Chapter 1, “Configuring Layer 3 Virtualization”	Describes how to configure Layer 3 virtualization.
Chapter 1, “Managing the Unicast RIB and FIB”	Describes how to view and modify the unicast RIB and FIB.
Chapter 1, “Configuring Route Policy Manager”	Describes how to configure the Route Policy Manager, including IP prefix lists and route maps for filtering and redistribution.
Chapter 1, “Configuring Policy Based Routing”	Describes how to configure Policy-Based Routing and includes guidelines, limitations, and examples.
Chapter 1, “Configuring HSRP”	Describes how to configure the Hot Standby Routing Protocol.
Chapter 1, “Configuring VRRP”	Describes how to configure the Virtual Router Redundancy Protocol.
Chapter 1, “Configuring Object Tracking”	Describes how to configure object tracking.
Appendix 1, “IETF RFCs supported by Cisco NX-OS Unicast Features, Release 6.x”	Lists IETF RFCs supported by Cisco NX-OS.

Document Conventions

Command descriptions use these conventions:

Convention	Description
boldface font	Commands and keywords are in boldface.
italic font	Arguments for which you supply values are in italics.
[ ]	Elements in square brackets are optional.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.

Screen examples use these conventions:

`screen font`	Terminal sessions and information that 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.

This document uses the following conventions:

Note Means reader *take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Related Documentation

Documentation for Cisco Nexus 6000 Switches and Cisco Nexus 2000 Series Fabric Extender is available at the following URL:

http://www.cisco.com/en/US/products/ps9670/tsd_products_support_series_home.html

The following are related Cisco Nexus 6000 Series and Cisco Nexus 2000 Series Fabric Extender documents:

Release Notes

Cisco Nexus 6000 Series and Cisco Nexus 2000 Series Release Notes

Cisco Nexus 6000 Series Switch Release Notes

Maintain and Operate Guides

Cisco Nexus 6000 Series NX-OS Operations Guide

Installation and Upgrade Guides

Cisco Nexus 6000 Series Platform Hardware Installation Guide

Cisco Nexus 2000 Series Hardware Installation Guide

Regulatory Compliance and Safety Information for the Cisco Nexus 6000 Series Switches and Cisco Nexus 2000 Series Fabric Extenders

Licensing Guide

Cisco NX-OS Licensing Guide

Command References

Cisco Nexus 6000 Series Command Reference

Error and System Messages

Cisco NX-OS System Messages Reference

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see *What's New in Cisco Product Documentation* at: http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html.

Subscribe to *What's New in Cisco Product Documentation*, which lists all new and revised Cisco technical documentation as an RSS feed and delivers content directly to your desktop using a reader application. The RSS feeds are a free service.

Overview

This chapter introduces the basic concepts for Layer 3 unicast routing protocols in Cisco NX-OS.

This chapter includes the following sections:

- Information About Layer 3 Unicast Routing, page 1-1
- Routing Algorithms, page 1-8
- Layer 3 Virtualization, page 1-10
- Cisco NX-OS Fowarding Architecture, page 1-10
- Summary of Layer 3 Unicast Routing Features, page 1-12
- Related Topics, page 1-14

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.

This section includes the following topics:

- Routing Fundamentals, page 1-2
- Packet Switching, page 1-2
- Routing Metrics, page 1-3
- Router IDs, page 1-5
- Autonomous Systems, page 1-5
- Convergence, page 1-6
- Load Balancing and Equal Cost Multipath, page 1-6
- Route Redistribution, page 1-6
- Administrative Distance, page 1-7
- Stub Routing, page 1-7

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 and the address of the next router or *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 on page 1-10 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 on page 1-3.

Routers communicate with one another and maintain their routing tables by transmitting a variety of messages. The routing update message is one of these messages 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 on page 1-8.

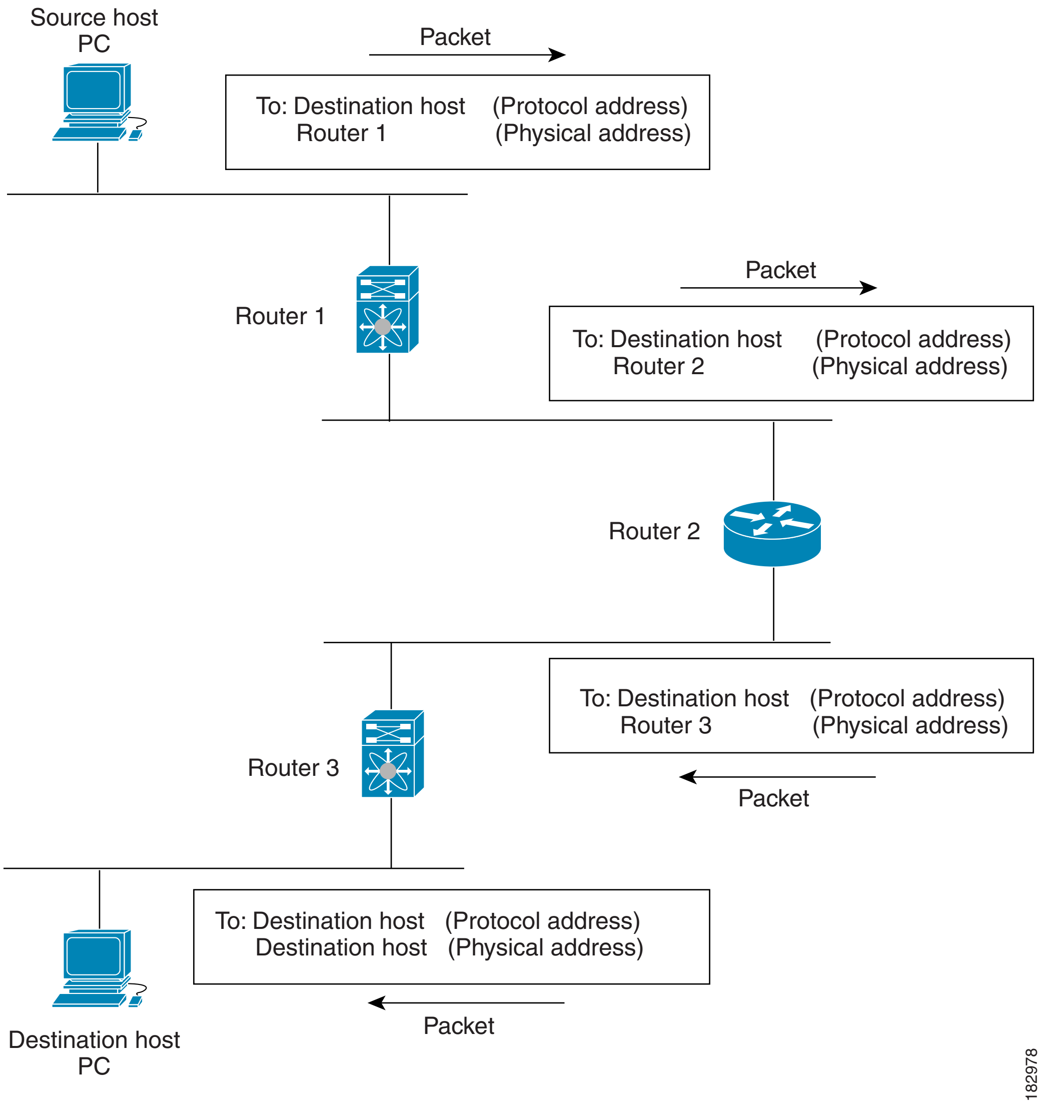
Packet Switching

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 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 Figure 1-1).

Figure 1-1 Packet Header Updates Through a Network


Source host
PC
Packet
To: Destination host (Protocol address)
Router 1 (Physical address)
Router 1
Packet
To: Destination host (Protocol address)
Router 2 (Physical address)
Router 2
To: Destination host (Protocol address)
Router 3 (Physical address)
Router 3
Packet
To: Destination host (Protocol address)
Destination host (Physical address)
Packet
Destination host
PC
182978

Routing Metrics

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

This section includes the following metrics:

- Path Length, page 1-4
- Reliability, page 1-4
- Routing Delay, page 1-4
- Bandwidth, page 1-4
- Load, page 1-4

- Communication Cost, page 1-4

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 hop count, 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 needs to 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 would be 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 utilization 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 any loopback interfaces, 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 later 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.

Autonomous Systems

An *autonomous system* (AS) is a network controlled by a single technical administration entity. Autonomous systems divide global external networks into individual routing domains, where local routing policies are applied. This organization simplifies routing domain administration and simplifies consistent policy configuration.

Each autonomous system can support multiple interior routing protocols that dynamically exchange routing information through route *redistribution*. The Regional Internet Registries assign a unique number to each public autonomous system that directly connects to the Internet. This autonomous system number (AS number) identifies both the routing process and the autonomous system.

Cisco NX-OS supports 4-byte AS numbers. Table 1-1 lists the AS number ranges.

Table 1-1 AS Numbers

2-Byte Numbers	4-Byte Numbers in AS.dot Notation	4-Byte Numbers in plaintext Notation	Purpose
1 to 64511	0.1 to 0.64511	1 to 64511	Public AS (assigned by RIR)[1]
64512 to 65534	0.64512 to 0.65534	64512 to 65534	Private AS (assigned by local administrator)
65535	0.65535	65535	Reserved
N/A	1.0 to 65535.65535	65536 to 4294967295	Public AS (assigned by RIR)

[1] RIR=Regional Internet Registries

Private autonomous system numbers are used for internal routing domains but must be translated by the router for traffic that is routed out to the Internet. You should not configure routing protocols to advertise private autonomous system numbers to external networks. By default, Cisco NX-OS does not remove private autonomous system numbers from routing updates.

Note The autonomous system number assignment for public and private networks is governed by the Internet Assigned Number Authority (IANA). For information about autonomous system numbers, including the reserved number assignment, or to apply to register an autonomous system number, refer to the following URL:
http://www.iana.org/

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, 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.

Load Balancing and Equal Cost Multipath

Routing protocols can use *load balancing* or equal cost multipath (ECMP) to share traffic across multiple paths.When a router learns multiple routes to a specific network, it installs the route with the lowest administrative distance in the routing table. If the router receives and installs multiple paths with the same administrative distance and cost to a destination, load balancing can occur. Load balancing distributes the traffic across all the paths, sharing the load. The number of paths used is limited by the number of entries that the routing protocol puts in the routing table.Cisco Nexus 5500 series switches support up to 16 paths and Cisco Nexus 6000 series switches support up to 64 paths to a destination for BGP, EIGRP, and OSPF.

The Enhanced Interior Gateway Routing Protocol (EIGRP) also supports unequal cost load balancing. For more information, see Chapter 1, “Configuring EIGRP.”

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 Open Shortest Path First (OSPF) 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. This avoids the problem of 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.

Route redistribution also uses an administrative distance (see the “Administrative Distance” section on page 1-7) 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 chosen 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. The higher the value, the lower the 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

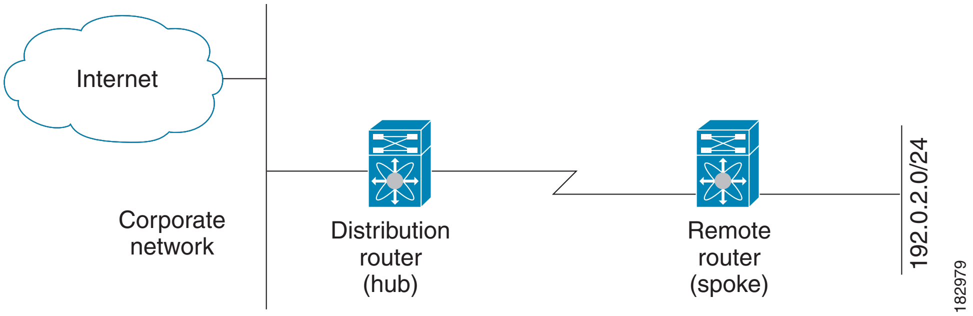
You can use stub routing in a hub-and-spoke network topology, where one or more end (stub) networks are connected to a remote router (the spoke) that is connected to one or more distribution routers (the hub). The remote router is adjacent only to one or more distribution routers. The only route for 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 informing 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.

Figure 1-2 shows a simple hub-and-spoke configuration.

Figure 1-2 Simple Hub-and-Spoke Network



Stub routing does not prevent routes from being advertised to the remote router. Figure 1-2 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 would always be through the distribution router. A larger route table would reduce 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 EIGRP supports stub routers.

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.

This section includes the following topics:

- Static Routes and Dynamic Routing Protocols, page 1-8
- Interior and Exterior Gateway Protocols, page 1-8
- Distance Vector Protocols, page 1-9
- Link-State Protocols, page 1-9

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 today's large, constantly changing networks. Most routing protocols today use dynamic routing algorithms, which 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. 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 feature 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), which 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 received to its neighbors to prevent an LSA update loop. Because the router floods the LSAs immediately after they receive them, 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 its 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 somewhat less prone to routing loops than distance vector algorithms. However, link-state algorithms require more CPU power and memory than distance vector algorithms. Link-state algorithms 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 Instances (VRFs) and multiple routing information bases (*RIB*s) 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 Chapter 1, "Configuring Layer 3 Virtualization."

Cisco NX-OS Fowarding Architecture

The Cisco NX-OS forwarding architecture is responsible for processing all routing updates and populating the forwarding information on the switch.

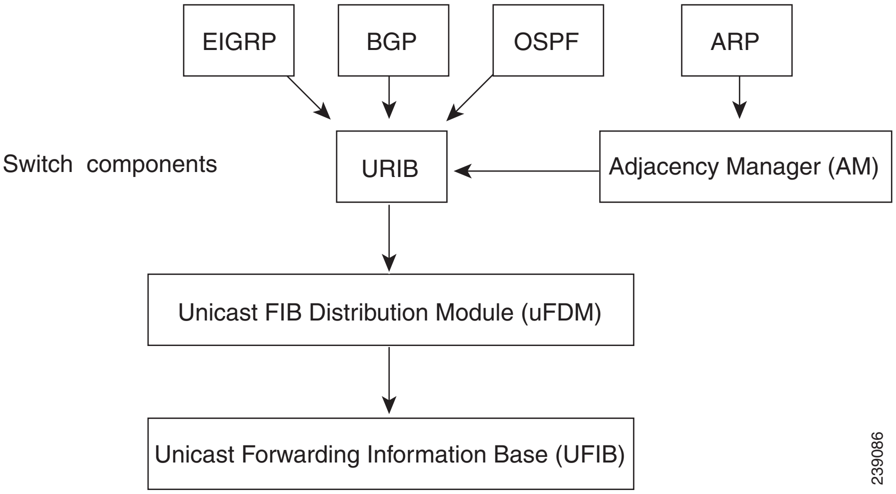
This section includes the following topics:

- Unicast RIB, page 1-10
- Adjacency Manager, page 1-11
- Unicast Forwarding Distribution Module, page 1-11
- FIB, page 1-11
- Hardware Forwarding, page 1-12
- Software Forwarding, page 1-12

Unicast RIB

The Cisco NX-OS forwarding architecture consists of multiple components, as shown in Figure 1-3.

Figure 1-3 Cisco NX-OS Forwarding Architecture


EIGRP
BGP
OSPF
ARP
Switch components
URIB
Adjacency Manager (AM)
Unicast FIB Distribution Module (uFDM)
Unicast Forwarding Information Base (UFIB)
239086

The unicast RIB 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 unicast forwarding information base (FIB) by using the services of 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 maintains adjacency information for different protocols including ARP, Open Shortest Path First version 2 (OSPFv2), Neighbor Discovery Protocol (NDP), 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. For IPv6, the adjacency manager finds the Layer 3 to Layer 2 mapping information from NDP. See Chapter 3, “Configuring IPv6.”

Unicast Forwarding Distribution Module

The unicast forwarding distribution module distributes the forwarding path information from the unicast RIB and other sources. The unicast RIB generates forwarding information which the unicast FIB programs into the hardware forwarding tables. The unicast forwarding distribution module also downloads the FIB information to newly inserted modules.

The unicast forwarding distribution module 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 builds the information used for the hardware forwarding engine. The unicast FIB receives route updates from the unicast forwarding distribution module and sends the information along 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 and one for IPv6 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.

The unicast FIB also enables and disables unicast reverse path forwarding (RPF) checks per interface. The Cisco Nexus 5548 switch supports the following two RPF modes that can be configured on each ingress interface:

- RPF Strict Check—Packets that do not have a verifiable source address in the routers forwarding table or do not arrive on any of the return paths to the source are dropped.
- RPF Loose Check—Packets have a verifiable source address in the routers forwarding table and the source is reachable through a physical interface. The ingress interface that receives the packet need not match any of the interfaces in the FIB.

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.

You also can use the **show platform fib** or **show platform forwarding** commands to display details on hardware forwarding.

Software Forwarding

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

Software forwarding is controlled by control plane policies and 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.

This section includes the following topics:

- IPv4 and IPv6, page 1-13
- OSPF, page 1-13
- OSPF, page 1-13
- EIGRP, page 1-13
- BGP, page 1-13
- RIP, page 1-13
- Static Routing, page 1-13
- Layer 3 Virtualization, page 1-14
- Route Policy Manager, page 1-14
- First-Hop Redundancy Protocols, page 1-14
- Object Tracking, page 1-14

IPv4 and IPv6

Layer 3 uses either the IPv4 or IPv6 protocol. IPv6 is a new IP protocol designed to replace IPv4, the Internet protocol that is predominantly deployed and used throughout the world. IPv6 increases the number of network address bits from 32 bits (in IPv4) to 128 bits. For more information, see Chapter 1, "Configuring IPv4." or Chapter 3, "Configuring IPv6."

OSPF

The 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 connected to the link. The advertisements that contain this link information are called link-state advertisements. For more information, see Chapter 1, "Configuring OSPFv2."

EIGRP

The EIGRP protocol is a unicast routing protocol that has the characteristics of both distance vector and link-state routing protocols. It is an improved version of IGRP, which is a Cisco proprietary routing protocol. EIGRP relies on its neighbors to provide the routes, typical to a distance vector routing protocol. It constructs the network topology from the routes advertised by its neighbors, similar to a link-state protocol, and uses this information to select loop-free paths to destinations. For more information, see Chapter 1, "Configuring EIGRP."

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 Chapter 1, "Configuring Basic BGP" and Chapter 1, "Configuring Advanced BGP."

RIP

The Routing Information Protocol (RIP) is a distance-vector protocol that uses a hop count as its metric. RIP is widely used for routing traffic in the global Internet and is an Interior Gateway Protocol (IGP), which means that it performs routing within a single autonomous system. For more information, see Chapter 1, "Configuring RIP."

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 Chapter 1, "Configuring Static Routing."

Layer 3 Virtualization

Virtualization allows you to share physical resources across separate management domains.

Cisco NX-OS supports Layer 3 virtualization with VPN Routing and Forwarding (VRF). A VRF provides a separate address domain for configuring Layer 3 routing protocols. For more information, see Chapter 1, "Configuring Layer 3 Virtualization."

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. For more information, see Chapter 1, "Configuring Route Policy Manager."

First-Hop Redundancy Protocols

A first-hop redundancy protocol (FHRP) allows 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 the Hot Standby Router Protocol (HSRP), see Chapter 1, "Configuring HSRP." For more information on the Virtual Router Redundancy Protocol (VRRP), see Chapter 1, "Configuring VRRP."

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 recovery time if an object state goes down. For more information, see Chapter 1, "Configuring Object Tracking."

Related Topics

The following Cisco documents are related to the Layer 3 features:

- *Cisco Nexus 6000 Series NX-OS Multicast Routing Configuration Guide, Release 6.0*
- Exploring Autonomous System Numbers:
  http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-1/autonomous_system_numbers.html

New and Changed Information

This chapter provides release-specific information for each new and changed feature in the *Cisco Nexus 6000 Series NX-OS Unicast Routing Configuration Guide, Release 6.x*. The latest version of this document is available at the following Cisco website:

http://www.cisco.com/en/US/products/ps9670/products_installation_and_configuration_guides_list.html

To check for additional information about Cisco NX-OS Release 5.x, see the *Cisco Nexus 6000 Series Switch NX-OS Release Notes* available at the following Cisco website:

http://www.cisco.com/en/US/products/ps9670/prod_release_notes_list.html

Table 1 summarizes the new and changed features for the *Cisco Nexus 6000 Series NX-OS Unicast Routing Configuration Guide, Release 6.x*, and tells you where they are documented.

Table 1 New and Changed Features for Release 6.x

Feature	Description	Changed in Release	Where Documented
Web Cache Communication Protocol (WCCP) v2	WCCPv2 specifies interactions between one or more Cisco NX-OS routers and one or more cache engines.	6.0(2)N3(1)	Chapter 4, "Configuring WCCPv2."
Bidirectional Forwarding Detection (BFD)	BFD was introduced for OSPF, BGP, EIGRP., Static Routes, PIM, VRRP, and HSRP.	6.0(2)N2(1)	Chapter 1, "Configuring OSPFv2,"BFD, page 1-11 Chapter 1, "Configuring EIGRP,"BFD, page 1-7 Chapter 1, "Configuring Advanced BGP," BFD, page 1-8 Chapter 1, "Configuring Static Routing," BFD, page 1-3 Chapter 1, "Configuring HSRP," BFD, page 1-7 Chapter 1, "Configuring VRRP," BFD, page 1-5
Policy-Based Routing	This feature was introduced.	6.0(2)N2(1)	Chapter 1, "Configuring Policy Based Routing"

Table 1 New and Changed Features for Release 6.x (continued)

Feature	Description	Changed in Release	Where Documented
ECMP maximum paths	For BGP, EIGRP, and OSPF , the number of maximum paths that can be load-balanced to a destination in equal-cost multi-path (ECMP) routing has increased from 16 to 64.	6.0(2)N2(1)	Chapter 1, "Overview," Load Balancing and Equal Cost Multipath, page 1-6 Chapter 4, "Configuring OSPFv2," Configuring Optional Parameters on an OSPFv2 Instance, page 1-15 Chapter 6, "Configuring EIGRP," Configuring Load Balancing in EIGRP, page 1-21 Chapter 5, "Configuring OSPFv3," Creating an OSPFv3 Instance, page 1-14 Chapter 8, "Configuring Advanced BGP," Configuring Load Sharing and ECMP, page 1-27
ACLs for ip-directed broadcast command	This feature was introduced.	6.0(2)N1(2)	Configuring IP Directed Broadcasts, page 1-14. (For other 6.0(2)N1(2) features, see the *Cisco Nexus 6000 Series Release Notes, Cisco NX-OS Release 6.x.*)
Cisco Nexus 6000 switch	Initial product release	6.0(2)N1(1)	

CHAPTER 1

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 switch.

This chapter includes the following sections:

Information About IPv4

You can configure IP on the switch 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 switch. An interface can have one primary IP address and multiple secondary addresses. All networking switches on an interface should share the same primary IP address because the packets that are generated by the switch always use the primary IPv4 address. Each IPv4 packet is based on the information from a source or destination IP address. See the "Multiple IPv4 Addresses" section on page 1-2.

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 in the Cisco NX-OS system is responsible for handling IPv4 packets, as well as the forwarding of IPv4 packets, which includes IPv4 unicast and multicast route lookup, reverse path forwarding (RPF) checks, software access control list (ACL) forwarding, and and policy-based routing (PBR). The IP feature also manages the network interface IP address configuration, duplicate address checks, static routes, and packet send and receive interface for IP clients.

This section includes the following topics:

- Multiple IPv4 Addresses, page 1-2
- Address Resolution Protocol, page 1-3
- ARP Caching, page 1-3
- Static and Dynamic Entries in the ARP Cache, page 1-4
- Devices That Do Not Use ARP, page 1-4
- Reverse ARP, page 1-4
- Reverse ARP, page 1-4
- Proxy ARP, page 1-5
- Local Proxy ARP, page 1-5
- ACLs for IP Directed Broadcast, page 1-6
- Glean Throttling, page 1-6
- ICMP, page 1-6
- Virtualization Support, page 1-6

Multiple IPv4 Addresses

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

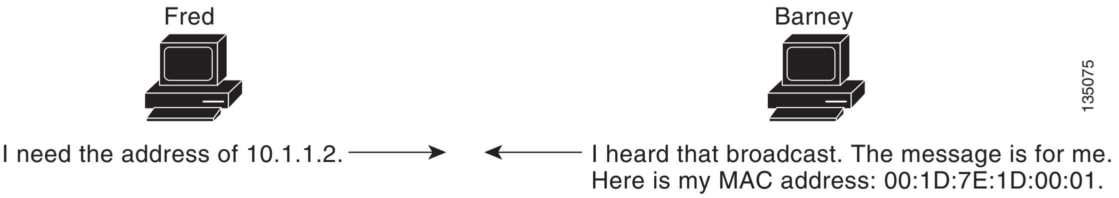
Note If any switch on a network segment uses a secondary IPv4 address, all other switches on that same network interface must also use a secondary address from the same network or subnet. The inconsistent use of secondary addresses on a network segment can quickly cause routing loops.

Address Resolution Protocol

Networking switches 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 switch sends a packet to another switch, it looks in its own ARP cache to see if there is a MAC address and corresponding IP address for the destination switch. If there is no entry, the source switch sends a broadcast message to every switch on the network.

Each switch compares the IP address to its own. Only the switch with the matching IP address replies to the switch that sends the data with a packet that contains the MAC address for the switch. The source switch adds the destination switch 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. Figure 1-1 shows the ARP broadcast and response process.

Figure 1-1 ARP Process


Fred
Barney
135075
I need the address of 10.1.1.2.
I heard that broadcast. The message is for me.
Here is my MAC address: 00:1D:7E:1D:00:01.

When the destination switch lies on a remote network which is beyond another switch, the process is the same except that the switch 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 switch on the destination switch network uses ARP to obtain the MAC address of the destination switch and delivers the packet. ARP is enabled by default.

The default system-defined CoPP policy rate-limits ARP broadcast packets. 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 (switch) 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 memory for a period of time, which minimizes the use of valuable network resources to broadcast for the same address each time a packet is sent. You must maintain the cache entries since the cache entries are set to expire periodically because the information might become outdated. Every switch on a network updates its tables as addresses are broadcast.

Static and Dynamic Entries in the ARP Cache

You must manually configure the IP addresses, subnet masks, gateways, and corresponding MAC addresses for each interface of each switch when using static routes. Static routing enables more control but 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 switches 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 that uses MAC addresses only, as opposed to a switch, which has an ARP cache that contains both IP addresses and the corresponding MAC addresses.

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

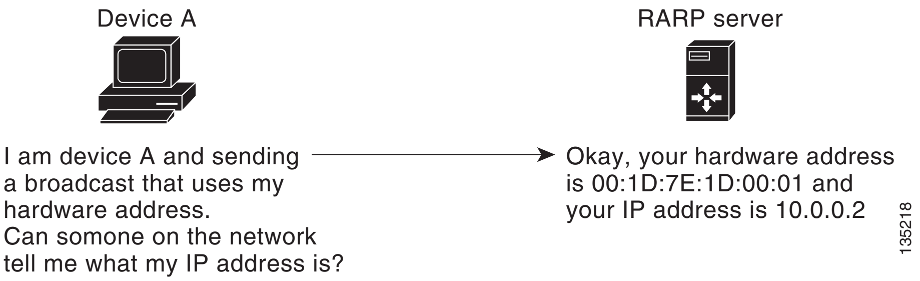
Layer 2 switches determine which port is connected to a device to which the message is addressed and send only to that port, unlike a hub, which sends the message out all of its ports. However, Layer 3 switches are switches that build an ARP cache (table).

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. Figure 1-2 illustrates how RARP works.

Figure 1-2 Reverse ARP


Device A
RARP server
I am device A and sending a broadcast that uses my hardware address. Can somone on the network tell me what my IP address is?
Okay, your hardware address is 00:1D:7E:1D:00:01 and your IP address is 10.0.0.2
135218

There are several limitations of RARP. Because of these limitations, most businesses use 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 switch that is physically located on one network appear to be logically part of a different physical network connected to the same switch or firewall. Proxy ARP allows you to hide a switch with a public IP address on a private network behind a router and still have the switch 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 switches on a subnet reach remote subnets without configuring routing or a default gateway.

When switches 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 switches 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 switch and it receives an ARP request, it identifies the request as a request for a system that is not on the local LAN. The switch responds as if it is the remote destination for which the broadcast is addressed, with an ARP response that associates the MAC address of the switch with the IP address of the remote destination. The local switch 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 switch. By default, Proxy ARP is disabled.

Local Proxy ARP

You can use local Proxy ARP to enable a switch 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 switch to which they are connected.

Gratuitous ARP

Gratuitous ARP sends a request with identical source IP address and destination IP address to detect duplicate IP addresses. Cisco NX-OS Release 5.0(3) support enabling or disabling gratuitous ARP requests or ARP cache updates.

ACLs for IP Directed Broadcast

You can use IP directed broadcast to broadcast to an IP subnet from a node that does not belong to it. You can specify an ACL list for the broadcast.

Glean Throttling

When forwarding an incoming IP packet in a line card, if the Address Resolution Protocol (ARP) request for the next hop is not resolved, the line card forwards the packets to the supervisor (glean throttling). The supervisor resolves the MAC address for the next hop and programs the hardware.

The Cisco Nexus 6000 Series device hardware has glean rate limiters to protect the supervisor from the glean traffic. If the maximum number of entries is exceeded, the packets for which the ARP request is not resolved continues to be processed in the software instead of getting dropped in the hardware.

When an ARP request is sent, the software adds a /32 drop adjacency in the hardware to prevent the packets to the same next-hop IP address to be forwarded to the supervisor. When the ARP is resolved, the hardware entry is updated with the correct MAC address. If the ARP entry is not resolved before a timeout period, the entry is removed from the hardware.

ICMP

You can use 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

IPv4 supports Virtual Routing and Forwarding instances (VRFs). By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. For more information, see Chapter 1, "Configuring Layer 3 Virtualization."

Licensing Requirements for IPv4

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	IPv4 requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for IPv4

IPv4 has the following prerequisites:

- IPv4 can only be configured on Layer 3 interfaces.

Guidelines and Limitations

IPv4 has the following configuration guidelines and limitations:

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

Default Settings

Table 1-1 lists the default settings for IP parameters.

Table 1-1 Default IP Parameters

Parameters	Default
ARP timeout	1500 seconds
proxy ARP	disabled

Configuring IPv4

This section includes the following topics:

- Configuring IPv4 Addressing, page 1-8
- Configuring Multiple IP Addresses, page 1-9
- Configuring a Static ARP Entry, page 1-10
- Configuring Proxy ARP, page 1-11
- Configuring Local Proxy ARP, page 1-12
- Configuring IP Directed Broadcasts, page 1-14
- Configuring IP Glean Throttling, page 1-15
- Configuring the Hardware IP Glean Throttle Maximum, page 1-16
- Configuring a Hardware IP Glean Throttle Timeout, page 1-17
- Verifying the IPv4 Configuration, page 1-18

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.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **no switchport**
4. **ip address** *ip-address/length* **[secondary]**
5. (Optional) **show ip interface**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface ethernet`** *`number`*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

	Command	Purpose
Step 4	**ip address** *ip-address/length* [**secondary**] **Example:** `switch(config-if)# ip address 192.2.1.1 255.0.0.0`	Specifies a primary or secondary IPv4 address for an interface. • 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. • The network mask can be indicated as a slash (/) and a number - a 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 is no space between the IP address and the slash.
Step 5	**show ip interface** **Example:** `switch(config-if)# show ip interface`	(Optional) Displays interfaces configured for IPv4.
Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to assign an IPv4 address:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip address 192.2.1.1 255.0.0.0
switch(config-if)# copy running-config startup-config
```

Configuring Multiple IP Addresses

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

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **no switchport**
4. **ip address** *ip-address/length* **[secondary]**
5. (Optional) **show ip interface**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface ethernet** *number*
	
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	`no switchport`
	
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	**ip address** *ip-address/length* [**secondary**] **Example:** `switch(config-if)# ip address 192.2.1.1 255.0.0.0 secondary`	Specifies the configured address as a secondary IPv4 address.

Step 5	`show ip interface`
	
	Example:
	`switch(config-if)# show ip interface`
		(Optional) Displays interfaces configured for IPv4.

Step 6	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring a Static ARP Entry

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

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **no switchport**
4. **ip arp** *ipaddr mac_addr*
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface ethernet** *number*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	`no switchport`
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	**ip arp** *ipaddr mac_addr* **Example:** `switch(config-if)# ip arp 192.2.1.1 0019.076c.1a78`	Associates an IP address with a MAC address as a static entry.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure a static ARP entry:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip arp 192.2.1.1 0019.076c.1a78
switch(config-if)# copy running-config startup-config
```

Configuring Proxy ARP

You can configure Proxy ARP on the switch to determine the media addresses of hosts on other networks or subnets.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **no switchport**
4. **ip proxy-arp**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface ethernet`** *`number`*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	`no switchport`
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	`ip proxy-arp`
	Example:
	`switch(config-if)# ip proxy-arp`
		Enables Proxy ARP on the interface.

Step 5	**`copy running-config startup-config`**
	Example:
	`switch(config-if)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to configure Proxy ARP:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip proxy-arp
switch(config-if)# copy running-config startup-config
```

Configuring Local Proxy ARP

You can configure Local Proxy ARP on the switch.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **no switchport**
4. **ip local-proxy-arp**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**interface ethernet** *number* **Example:** switch(config)# interface ethernet 2/3 switch(config-if)#	Enters interface configuration mode.
Step 3	no switchport **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 4	ip local-proxy-arp **Example:** switch(config-if)# ip local-proxy-arp	Enables Local Proxy ARP on the interface.
Step 5	**copy running-config startup-config** **Example:** switch(config-if)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure Local Proxy ARP:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# ip local-proxy-arp
switch(config-if)# copy running-config startup-config
```

Configuring Gratuitous ARP

You can configure gratuitous ARP on an interface.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **no switchport**
4. **ip arp gratuitous** {**request | update**}
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface ethernet** *number*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	no switchport **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.

Step 4	**ip arp gratuitous** {**request** | **update**}
	Example:
	`switch(config-if)# ip arp gratuitous request`
		Enables gratuitous ARP on the interface. Default is enabled.

Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to disable gratuitous ARP requests:

```
switch# configure terminal
switch(config)# interface ethernet 2/3
switch(config-if)# no switchport
switch(config-if)# no ip arp gratuitous request
switch(config-if)# copy running-config startup-config
```

Configuring IP Directed Broadcasts

An IP directed broadcast is an IP packet whose destination address is a valid broadcast address for some IP subnet, but which originates from a node that is not itself part of that destination subnet.

A switch that is not directly connected to its destination subnet forwards an IP directed broadcast in the same way it would forward unicast IP packets destined to a host on that subnet. When a directed broadcast packet reaches a switch that is directly connected to its destination subnet, that packet is "exploded" as a broadcast on the destination subnet. The destination address in the IP header of the packet is rewritten to the configured IP broadcast address for the subnet, and the packet is sent as a link-layer broadcast.

If directed broadcast is enabled for an interface, incoming IP packets whose addresses identify them as directed broadcasts intended for the subnet to which that interface is attached will be exploded as broadcasts on that subnet.

To enable IP directed broadcasts, use the following command in interface configuration mode:

Command	Purpose
ip directed-broadcast *[acl-name]*	Enables the translation of a directed broadcast to physical broadcasts. An Access Control List (ACL) name may be specified. The name is a case-sensitive alphanumeric string up to 63 characters long.

Configuring IP Glean Throttling

Cisco NX-OS software supports glean throttling rate limiters to protect the supervisor from the glean traffic.

You can enable IP glean throttling.

Note We recommend that you configure the IP glean throttle feature by using the **hardware ip glean throttle** command to filter the unnecessary glean packets that are sent to the supervisor for ARP resolution for the next hops that are not reachable or do not exist. IP glean throttling boosts software performance and helps to manage traffic more efficiently.

SUMMARY STEPS

1. **configure terminal**
2. **hardware ip glean throttle**
3. **no hardware ip glean throttle**
4. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`hardware ip glean throttle`**
	Example:
	`switch(config)# hardware ip glean throttle`
		Enables ARP throttling.

	Command	Purpose

Step 3	`no` **`hardware ip glean throttle`**
	`Example:`
	`switch(config)# no hardware ip glean throttle`
		Disables ARP throttling.

Step 4	**`copy running-config startup-config`**
	`Example:`
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to enable IP glean throttling:

```
switch# configure terminal
switch(config)# hardware ip glean throttle
switch(config-if)# copy running-config startup-config
```

Configuring the Hardware IP Glean Throttle Maximum

You can limit the maximum number of drop adjacencies that are installed in the Forwarding Information Base (FIB).

SUMMARY STEPS

1. **configure terminal**
2. **hardware ip glean throttle maximum** *count*
3. **no hardware ip glean throttle maximum** *count*
4. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`hardware ip glean throttle maximum`** ***`count`***
	`Example:`
	`switch(config)# hardware ip glean throttle maximum 2134`
		Configures the number of drop adjacencies that are installed in the FIB.

	Command	Purpose
Step 3	**no hardware ip glean throttle maximum** ***count*** **Example:** `switch(config)# no hardware ip glean throttle maximum 2134`	Applies the default limits. The default value is 1000. The range is from 0 to 4095 entries.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to limit the maximum number of drop adjacencies that are installed in the FIB:

```
switch# configure terminal
switch(config)# hardware ip glean throttle maximum 2134
switch(config-if)# copy running-config startup-config
```

Configuring a Hardware IP Glean Throttle Timeout

You can configure a timeout for the installed drop adjacencies to remain in the FIB.

SUMMARY STEPS

1. **configure terminal**
2. **hardware ip glean throttle maximum timeout** ***timeout-in-sec***
3. **no hardware ip glean throttle maximum timeout** ***timeout-in-sec***
4. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**hardware ip glean throttle maximum timeout** ***timeout-in-sec*** **Example:** `switch(config)# hardware ip glean throttle maximum timeout 300`	Configures the timeout for the installed drop adjacencies to remain in the FIB.

	Command	Purpose
Step 3	**no hardware ip glean throttle maximum timeout** ***timeout-in-sec*** **Example:** `switch(config)# no hardware ip glean throttle maximum timeout 300`	Applies the default limits. The timeout value is in seconds. The range is from 300 seconds (5 minutes) to 1800 seconds (30 minutes). **Note** After the timeout period is exceeded, the drop adjacencies are removed from the FIB.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure a timeout for the drop adjacencies that are installed.

```
switch# configure terminal
switch(config)# hardware ip glean throttle maximum timeout 300
switch(config-if)# copy running-config startup-config
```

Verifying the IPv4 Configuration

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

Command	Purpose
show hardware forwarding ip verify	Displays the IP packet verification configuration.
show ip adjacency	Displays the adjacency table.
show ip arp	Displays the ARP table.
show ip interface	Displays IP-related interface information.
show ip arp statistics [**vrf** *vrf-name*]	Displays the ARP statistics.

Configuration Examples for IPv4

This example shows how to configure an IPv4 address:

```
configure terminal
 interface ethernet 1/2
  no switchport
  ip address 192.2.1.1/16
```

Additional References

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

- Related Documents, page 1-19
- Standards, page 1-19

Related Documents

Related Topic	Document Title
IP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

CHAPTER 1

Configuring IPv6

This chapter describes how to configure Internet Protocol version 6 (IPv6), which includes addressing, Neighbor Discovery Protocol (ND), and Internet Control Message Protocol version 6 (ICMPv6), on the Cisco NX-OS device.

This chapter includes the following sections:

Information About IPv6

IPv6, which is designed to replace IPv4, increases the number of network address bits from 32 bits (in IPv4) to 128 bits. IPv6 is based on IPv4 but it includes a much larger address space and other improvements such as a simplified main header and extension headers.

The larger IPv6 address space allows networks to scale and provide global reachability. The simplified IPv6 packet header format handles packets more efficiently. The flexibility of the IPv6 address space reduces the need for private addresses and the use of Network Address Translation (NAT), which translates private (not globally unique) addresses into a limited number of public addresses. IPv6 enables new application protocols that do not require special processing by border routers at the edge of networks.

IPv6 functionality, such as prefix aggregation, simplified network renumbering, and IPv6 site multihoming capabilities, enable more efficient routing. IPv6 supports Open Shortest Path First (OSPF) for IPv6 and multiprotocol Border Gateway Protocol (BGP).

This section includes the following topics:

- IPv6 Address Formats, page 1-2
- IPv6 Unicast Addresses, page 1-3

IPv6 Address Formats

An IPv6 address has 128 bits or 16 bytes. The address is divided into eight, 16-bit hexadecimal blocks separated by colons (:) in the format: x:x:x:x:x:x:x:x. Two examples of IPv6 addresses are as follows:

```
2001:0DB8:7654:3210:FEDC:BA98:7654:3210
2001:0DB8:0:0:8:800:200C:417A
```

IPv6 addresses contain consecutive zeros within the address. You can use two colons (::) at the beginning, middle, or end of an IPv6 address to replace the consecutive zeros. Table 1-1 shows a list of compressed IPv6 address formats.

Note You can use two colons (::) only once in an IPv6 address to replace the longest string of consecutive zeros within the address.

You can use a double colon as part of the IPv6 address when consecutive 16-bit values are denoted as zero. You can configure multiple IPv6 addresses per interface but only one link-local address.

The hexadecimal letters in IPv6 addresses are not case sensitive.

Table 1-1 Compressed IPv6 Address Formats

IPv6 Address Type	Preferred Format	Compressed Format
Unicast	2001:0:0:0:0DB8:800:200C:417A	2001::0DB8:800:200C:417A
Multicast	FF01:0:0:0:0:0:0:101	FF01::101
Loopback	0:0:0:0:0:0:0:1	::1
Unspecified	0:0:0:0:0:0:0:0	::

A node may use the loopback address listed in Table 1-1 to send an IPv6 packet to itself. The loopback address in IPv6 is the same as the loopback address in IPv4. For more information, see Chapter 1, "Overview."

Note You cannot assign the IPv6 loopback address to a physical interface. A packet that contains the IPv6 loopback address as its source or destination address must remain within the node that created the packet. IPv6 routers do not forward packets that have the IPv6 loopback address as their source or destination address.

Note You cannot assign an IPv6 unspecified address to an interface. You should not use the unspecified IPv6 addresses as destination addresses in IPv6 packets or the IPv6 routing header.

The IPv6 prefix is in the form documented in RFC 2373 where the IPv6 address is specified in hexadecimal using 16-bit values between colons. 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). For example, 2001:0DB8:8086:6502::/32 is a valid IPv6 prefix.

IPv6 Unicast Addresses

An IPv6 unicast address is an identifier for a single interface on a single node. A packet that is sent to a unicast address is delivered to the interface identified by that address. This section includes the following topics:

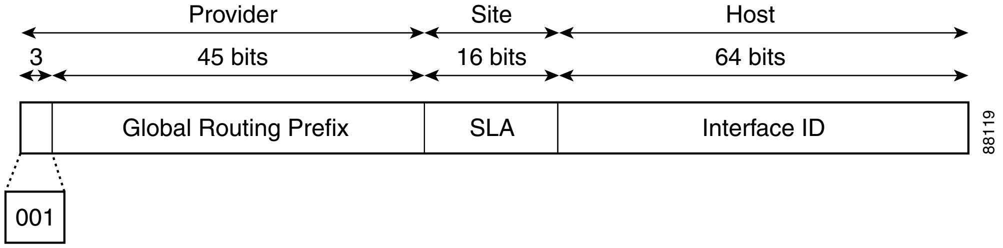
- Aggregatable Global Addresses, page 1-3
- Link-Local Addresses, page 1-5
- IPv4-Compatible IPv6 Addresses, page 1-5
- Unique Local Addresses, page 1-6
- Site-Local Address, page 1-6

Aggregatable Global Addresses

An aggregatable global address is an IPv6 address from the aggregatable global unicast prefix. The structure of aggregatable global unicast addresses enables strict aggregation of routing prefixes that limits the number of routing table entries in the global routing table. Aggregatable global addresses are used on links that are aggregated upward through organizations and eventually to the Internet service providers (ISPs).

Aggregatable global IPv6 addresses are defined by a global routing prefix, a subnet ID, and an interface ID. Except for addresses that start with binary 000, all global unicast addresses have a 64-bit interface ID. The IPv6 global unicast address allocation uses the range of addresses that start with binary value 001 (2000::/3). Figure 1-1 shows the structure of an aggregatable global address.

Figure 1-1 Aggregatable Global Address Format



Addresses with a prefix of 2000::/3 (001) through E000::/3 (111) are required to have 64-bit interface identifiers in the extended universal identifier (EUI)-64 format. The Internet Assigned Numbers Authority (IANA) allocates the IPv6 address space in the range of 2000::/16 to regional registries.

The aggregatable global address consists of a 48-bit global routing prefix and a 16-bit subnet ID or Site-Level Aggregator (SLA). In the IPv6 aggregatable global unicast address format document (RFC 2374), the global routing prefix included two other hierarchically structured fields called Top-Level Aggregator (TLA) and Next-Level Aggregator (NLA). The IETF decided to remove the TLS and NLA fields from the RFCs because these fields are policy based. Some existing IPv6 networks deployed before the change might still use networks that are on the older architecture.

A subnet ID, which is a 16-bit subnet field, can be used by individual organizations to create a local addressing hierarchy and to identify subnets. A subnet ID is similar to a subnet in IPv4, except that an organization with an IPv6 subnet ID can support up to 65,535 individual subnets.

An interface ID identifies interfaces on a link. The interface ID is unique to the link. In many cases, an interface ID is the same as or based on the link-layer address of an interface. Interface IDs used in aggregatable global unicast and other IPv6 address types have 64 bits and are in the modified EUI-64 format.

Interface IDs are in the modified EUI-64 format in one of the following ways:

- For all IEEE 802 interface types (for example, Ethernet, and Fiber Distributed Data interfaces), the first three octets (24 bits) are the Organizationally Unique Identifier (OUI) of the 48-bit link-layer address (MAC address) of the interface, the fourth and fifth octets (16 bits) are a fixed hexadecimal value of FFFE, and the last three octets (24 bits) are the last three octets of the MAC address. The Universal/Local (U/L) bit, which is the seventh bit of the first octet, has a value of 0 or 1. Zero indicates a locally administered identifier; 1 indicates a globally unique IPv6 interface identifier.
- For all other interface types (for example, serial, loopback, ATM, Frame Relay types—the interface ID is similar to the interface ID for IEEE 802 interface types; however, the first MAC address from the pool of MAC addresses in the router is used as the identifier because the interface does not have a MAC address.

Note For interfaces that use the Point-to-Point Protocol (PPP), where the interfaces at both ends of the connection might have the same MAC address, the interface identifiers at both ends of the connection are negotiated (picked randomly and, if necessary, reconstructed) until both identifiers are unique. The first MAC address in the router is used as the identifier for interfaces using PPP.

If no IEEE 802 interface types are in the router, link-local IPv6 addresses are generated on the interfaces in the router in the following sequence:

1. The router is queried for MAC addresses (from the pool of MAC addresses in the router).

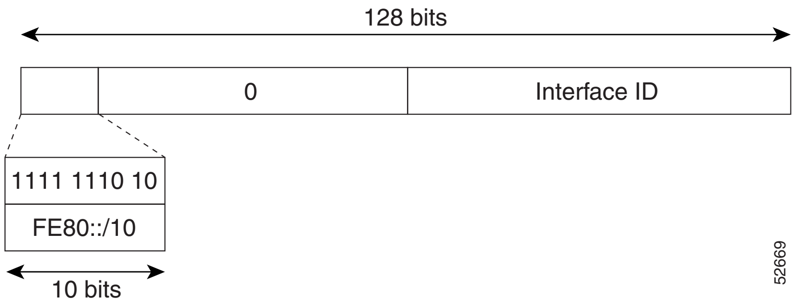
2. If no MAC addresses are available in the router, the serial number of the router is used to form the link-local addresses.
3. If the serial number of the router cannot be used to form the link-local addresses, the router uses a Message Digest 5 (MD5) hash to determine the MAC address of the router from the hostname of the router.

Link-Local Addresses

A link-local address is an IPv6 unicast address that can be automatically configured on any interface using the link-local prefix FE80::/10 (1111 1110 10) and the interface identifier in the modified EUI-64 format. Link-local addresses are used in the Neighbor Discovery Protocol (NDP) and the stateless autoconfiguration process. Nodes on a local link can use link-local addresses to communicate; the nodes do not need globally unique addresses to communicate. Figure 1-2 shows the structure of a link-local address.

IPv6 routers cannot forward packets that have link-local source or destination addresses to other links.

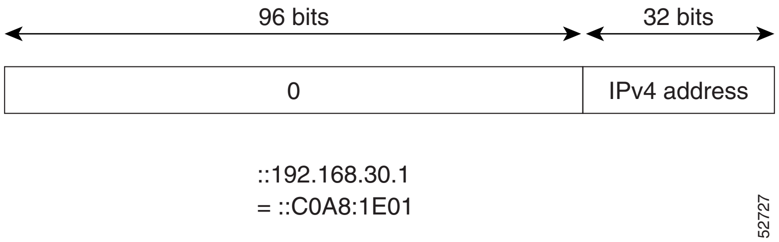
Figure 1-2 Link-Local Address Format



IPv4-Compatible IPv6 Addresses

An IPv4-compatible IPv6 address is an IPv6 unicast address that has zeros in the high-order 96 bits of the address and an IPv4 address in the low-order 32 bits of the address. The format of an IPv4-compatible IPv6 address is 0:0:0:0:0:0:A.B.C.D or ::A.B.C.D. The entire 128-bit IPv4-compatible IPv6 address is used as the IPv6 address of a node and the IPv4 address embedded in the low-order 32 bits is used as the IPv4 address of the node. IPv4-compatible IPv6 addresses are assigned to nodes that support both the IPv4 and IPv6 protocol stacks and are used in automatic tunnels. Figure 1-3 shows the structure of an IPv4-compatible IPv6 address and a few acceptable formats for the address.

Figure 1-3 IPv4-Compatible IPv6 Address Format



Unique Local Addresses

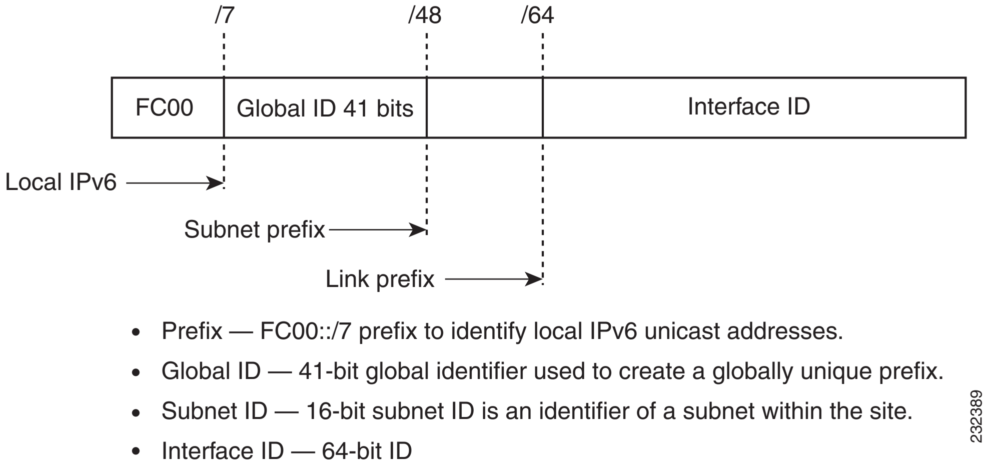
A unique local address is an IPv6 unicast address that is globally unique and is intended for local communications. It is not expected to be routable on the global Internet and is routable inside of a limited area, such as a site, and it may be routed between a limited set of sites. Applications may treat unique local addresses like global scoped addresses.

A unique local address has the following characteristics:

- It has a globally unique prefix (it has a high probability of uniqueness).
- It has a well-known prefix to allow for easy filtering at site boundaries.
- It allows sites to be combined or privately interconnected without creating any address conflicts or requiring renumbering of interfaces that use these prefixes.
- It is ISP-independent and can be used for communications inside of a site without having any permanent or intermittent Internet connectivity.
- If it is accidentally leaked outside of a site through routing or the Domain Name Server (DNS), there is no conflict with any other addresses.

Figure 1-4 shows the structure of a unique local address.

Figure 1-4 Unique Local Address Structure


/7
/48
/64
FC00
Global ID 41 bits
Interface ID
Local IPv6
Subnet prefix
Link prefix
• Prefix — FC00::/7 prefix to identify local IPv6 unicast addresses.
• Global ID — 41-bit global identifier used to create a globally unique prefix.
• Subnet ID — 16-bit subnet ID is an identifier of a subnet within the site.
• Interface ID — 64-bit ID
232389

Site-Local Address

Because RFC 3879 deprecates the use of site-local addresses, you should follow the recommendations of unique local addressing (ULA) in RFC 4193 when you configure private IPv6 addresses.

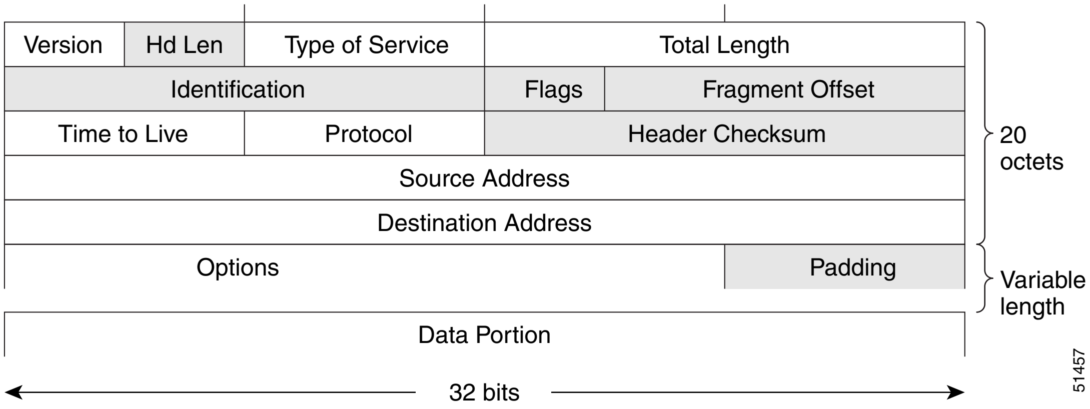
IPv6 Anycast Addresses

An anycast address is an address that is assigned to a set of interfaces that belong to different nodes. A packet sent to an anycast address is delivered to the closest interface—as defined by the routing protocols in use—identified by the anycast address. Anycast addresses are syntactically indistinguishable from unicast addresses because anycast addresses are allocated from the unicast address space. Assigning a unicast address to more than one interface turns a unicast address into an anycast address. You must configure the nodes to which the anycast address can recognize that the address is an anycast address.

Note Anycast addresses can be used only by a router, not a host. Anycast addresses cannot be used as the source address of an IPv6 packet.

Figure 1-5 shows the format of the subnet router anycast address; the address has a prefix concatenated by a series of zeros (the interface ID). The subnet router anycast address can be used to reach a router on the link that is identified by the prefix in the subnet router anycast address.

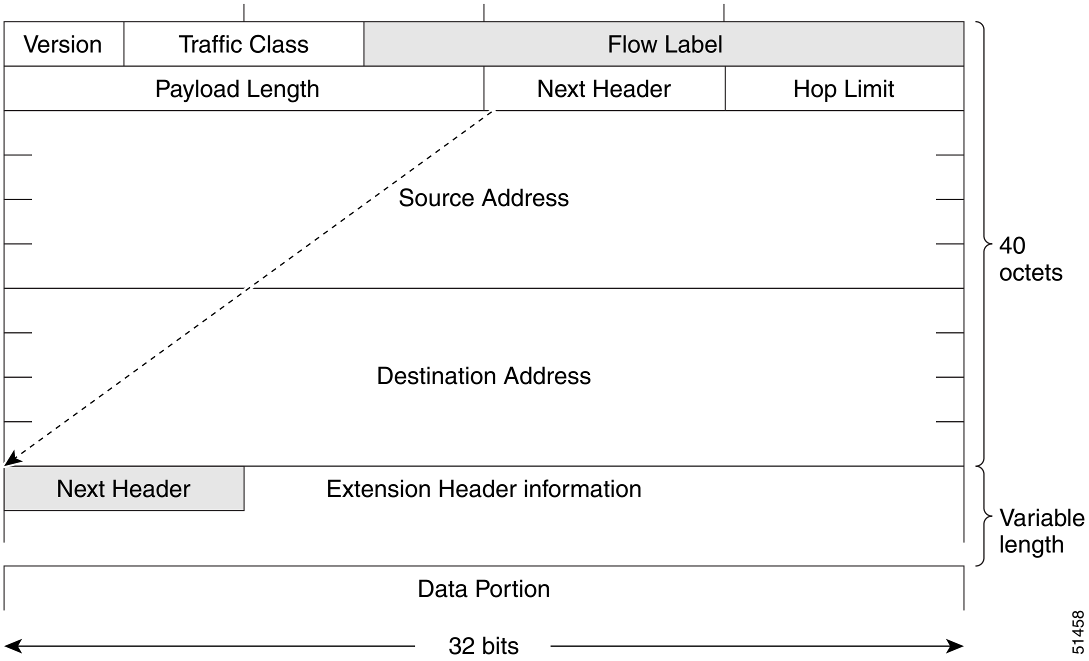
Figure 1-5 IPv4 Packet Header Format



IPv6 Multicast Addresses

An IPv6 multicast address is an IPv6 address that has a prefix of FF00::/8 (1111 1111). An IPv6 multicast address is an identifier for a set of interfaces that belong to different nodes. A packet sent to a multicast address is delivered to all interfaces identified by the multicast address. The second octet following the prefix defines the lifetime and scope of the multicast address. A permanent multicast address has a lifetime parameter equal to 0; a temporary multicast address has a lifetime parameter equal to 1. A multicast address that has the scope of a node, link, site, organization, or a global scope, has a scope parameter of 1, 2, 5, 8, or E, respectively. For example, a multicast address with the prefix FF02::/16 is a permanent multicast address with a link scope. Figure 1-6 shows the format of the IPv6 multicast address.

Figure 1-6 IPv6 Packet Header Format



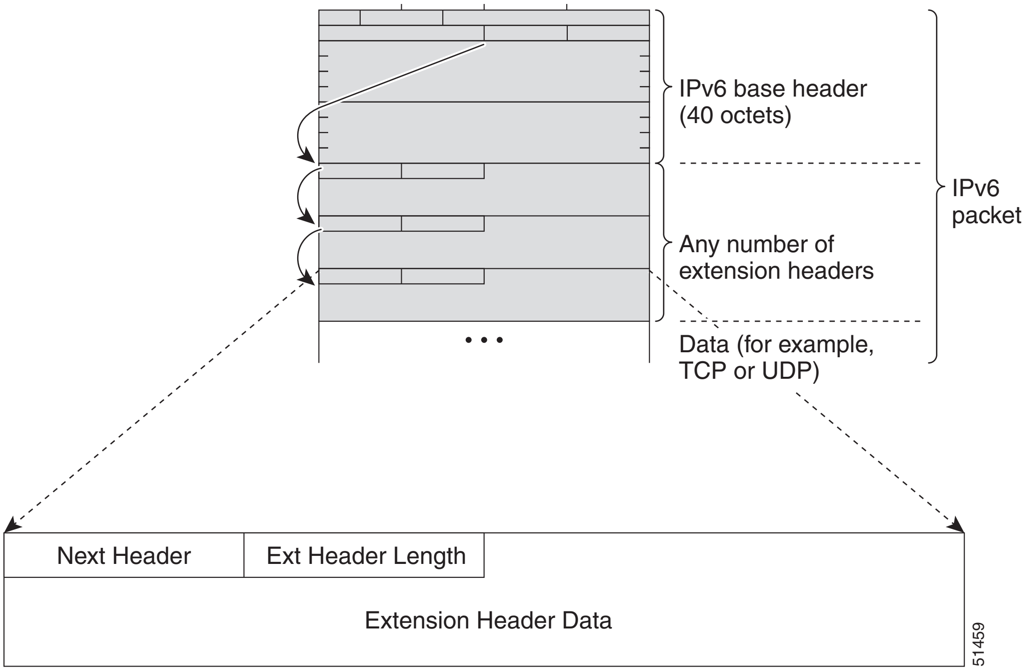
IPv6 nodes (hosts and routers) are required to join (where received packets are destined for) the following multicast groups:

- All-nodes multicast group FF02:0:0:0:0:0:0:1 (the scope is link-local)
- Solicited-node multicast group FF02:0:0:0:0:1:FF00:0000/104 for each of its assigned unicast and anycast addresses

IPv6 routers must also join the all-routers multicast group FF02:0:0:0:0:0:0:2 (the scope is link-local).

The solicited-node multicast address is a multicast group that corresponds to an IPv6 unicast or anycast address. IPv6 nodes must join the associated solicited-node multicast group for every unicast and anycast address to which it is assigned. The IPv6 solicited-node multicast address has the prefix FF02:0:0:0:0:1:FF00:0000/104 concatenated with the 24 low-order bits of a corresponding IPv6 unicast or anycast address (see Figure 1-7). For example, the solicited-node multicast address that corresponds to the IPv6 address 2037::01:800:200E:8C6C is FF02::1:FF0E:8C6C. Solicited-node addresses are used in neighbor solicitation messages.

Figure 1-7 IPv6 Extension Header Format

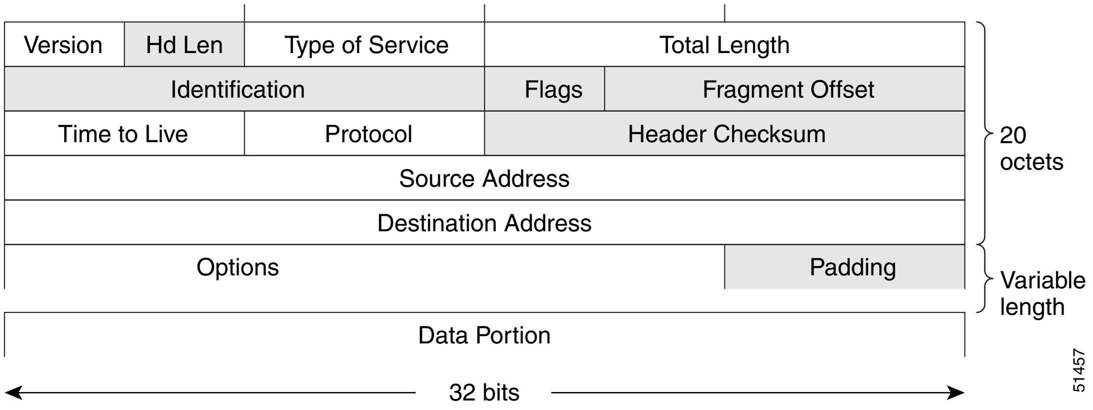


Note IPv6 has no broadcast addresses. IPv6 multicast addresses are used instead of broadcast addresses.

IPv4 Packet Header

The base IPv4 packet header has 12 fields with a total size of 20 octets (160 bits) (see Figure 1-5). The 12 fields may be followed by an Options field, which is followed by a data portion that is usually the transport-layer packet. The variable length of the Options field adds to the total size of the IPv4 packet header. The shaded fields of the IPv4 packet header are not included in the IPv6 packet header.

Figure 1-8 IPv4 Packet Header Format



Simplified IPv6 Packet Header

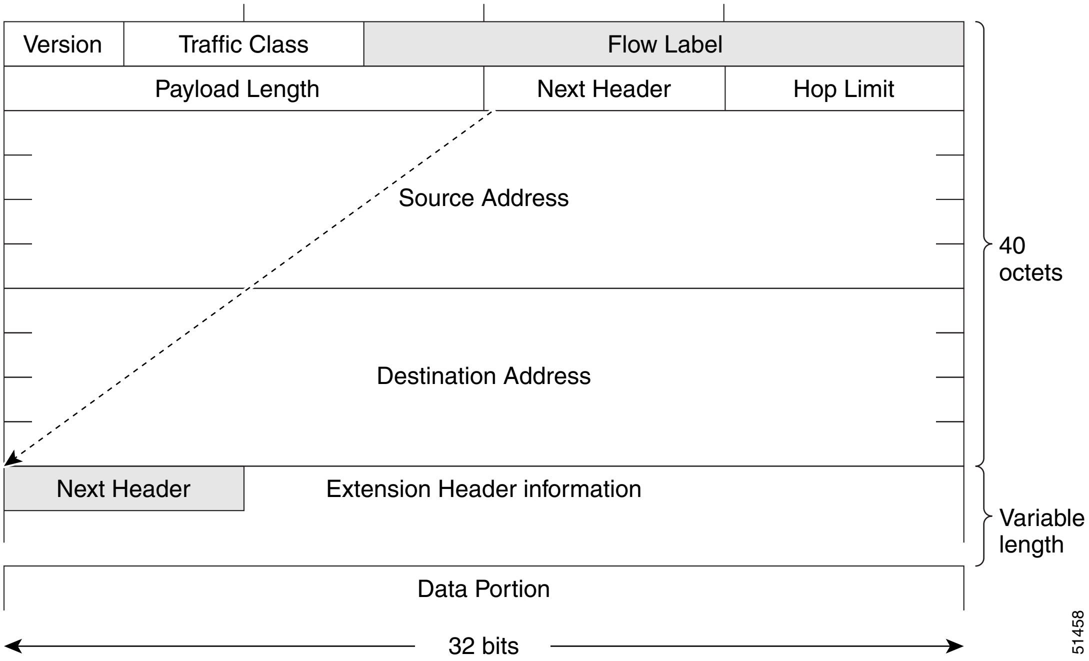
The base IPv6 packet header has 8 fields with a total size of 40 octets (320 bits) (see Figure 1-6). Fragmentation is handled by the source of a packet and checksums at the data link layer and transport layer are used. The User Datagram Protocol (UDP) checksum checks the integrity of the inner packet and the base IPv6 packet header and Options field are aligned to 64 bits, which can facilitate the processing of IPv6 packets.

Table 1-2 lists the fields in the base IPv6 packet header.

Table 1-2 Base IPv6 Packet Header Fields

Field	Description
Version	Similar to the Version field in the IPv4 packet header, except that the field lists number 6 for IPv6 instead of number 4 for IPv4.
Traffic Class	Similar to the Type of Service field in the IPv4 packet header. The Traffic Class field tags packets with a traffic class that is used in differentiated services.
Flow Label	New field in the IPv6 packet header. The Flow Label field tags packets with a specific flow that differentiates the packets at the network layer.
Payload Length	Similar to the Total Length field in the IPv4 packet header. The Payload Length field indicates the total length of the data portion of the packet.
Next Header	Similar to the Protocol field in the IPv4 packet header. The value of the Next Header field determines the type of information that follows the base IPv6 header. The type of information that follows the base IPv6 header can be a transport-layer packet, for example, a TCP or UDP packet, or an Extension Header, as shown in Figure 1-6.
Hop Limit	Similar to the Time to Live field in the IPv4 packet header. The value of the Hop Limit field specifies the maximum number of routers that an IPv6 packet can pass through before the packet is considered invalid. Each router decrements the value by one. Because no checksum is in the IPv6 header, the router can decrement the value without needing to recalculate the checksum, which saves processing resources.
Source Address	Similar to the Source Address field in the IPv4 packet header, except that the field contains a 128-bit source address for IPv6 instead of a 32-bit source address for IPv4.
Destination Address	Similar to the Destination Address field in the IPv4 packet header, except that the field contains a 128-bit destination address for IPv6 instead of a 32-bit destination address for IPv4.

Figure 1-9 IPv6 Packet Header Format



Optional extension headers and the data portion of the packet are after the eight fields of the base IPv6 packet header. If present, each extension header is aligned to 64 bits. There is no fixed number of extension headers in an IPv6 packet. Each extension header is identified by the Next Header field of the previous header. Typically, the final extension header has a Next Header field of a transport-layer protocol, such as TCP or UDP. Figure 1-7 shows the IPv6 extension header format.

Figure 1-10 IPv6 Extension Header Format

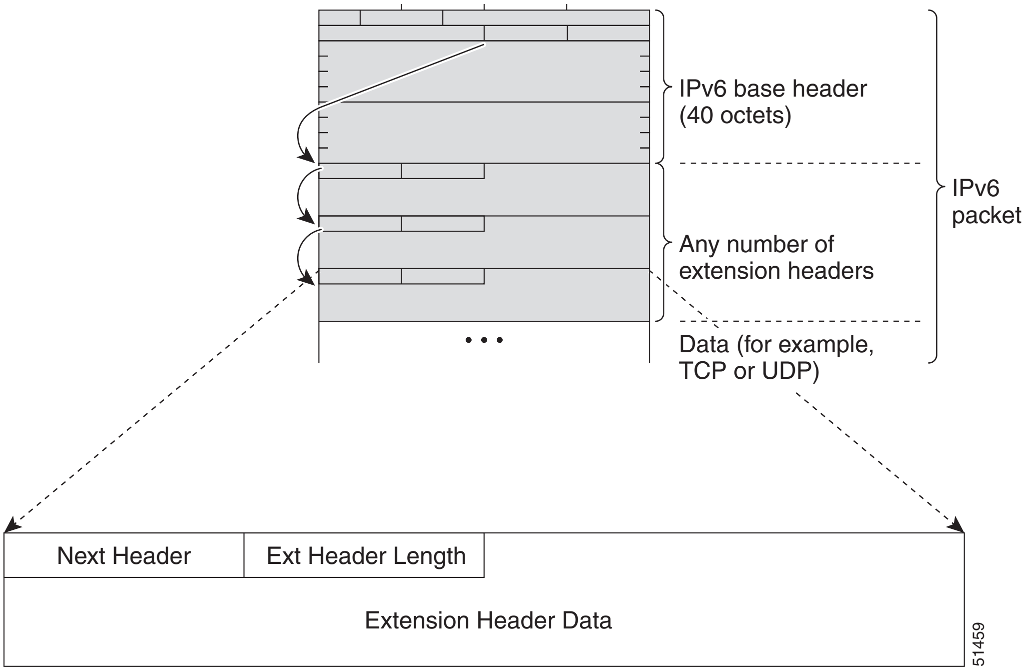

IPv6 base header (40 octets)
Any number of extension headers
IPv6 packet
Data (for example, TCP or UDP)
• • •
Next Header
Ext Header Length
Extension Header Data
51459

Table 1-3 lists the extension header types and their Next Header field values.

Table 1-3 IPv6 Extension Header Types

Header Type	Next Header Value	Description
Hop-by-hop options header	0	Header that is processed by all hops in the path of a packet. When present, the hop-by-hop options header always follows immediately after the base IPv6 packet header.
Destination options header	60	Header that can follow any hop-by-hop options header. The header is processed at the final destination and at each visited address specified by a routing header. Alternatively, the destination options header can follow any Encapsulating Security Payload (ESP) header. The destination options header is processed only at the final destination.
Routing header	43	Header that is used for source routing**.**
Fragment header	44	Header that is used when a source fragments a packet that is larger than the maximum transmission unit (MTU) for the path between itself and a destination. The Fragment header is used in each fragmented packet.
Upper-layer headers	6 (TCP) 17 (UDP)	Headers that are used inside a packet to transport the data. The two main transport protocols are TCP and UDP.

Path MTU Discovery for IPv6

As in IPv4, you can use path MTU discovery in IPv6 to allow a host to dynamically discover and adjust to differences in the MTU size of every link along a data path. In IPv6, however, fragmentation is handled by the source of a packet when the path MTU of one link along a given data path is not large enough to accommodate the size of the packets. Having IPv6 hosts handle packet fragmentation saves IPv6 router processing resources and helps IPv6 networks run more efficiently. Once the path MTU is reduced by the arrival of an ICMP Too Big message, Cisco NX-OS retains the lower value. The connection does not increase the segment size to gauge the throughput.

Note In IPv6, the minimum link MTU is 1280 octets. We recommend that you use an MTU value of 1500 octets for IPv6 links.

CDP IPv6 Address Support

You can use the Cisco Discovery Protocol (CDP) IPv6 address support for the neighbor information feature to transfer IPv6 addressing information between two Cisco devices. Cisco Discovery Protocol support for IPv6 addresses provides IPv6 information to network management products and troubleshooting tools.

ICMP for IPv6

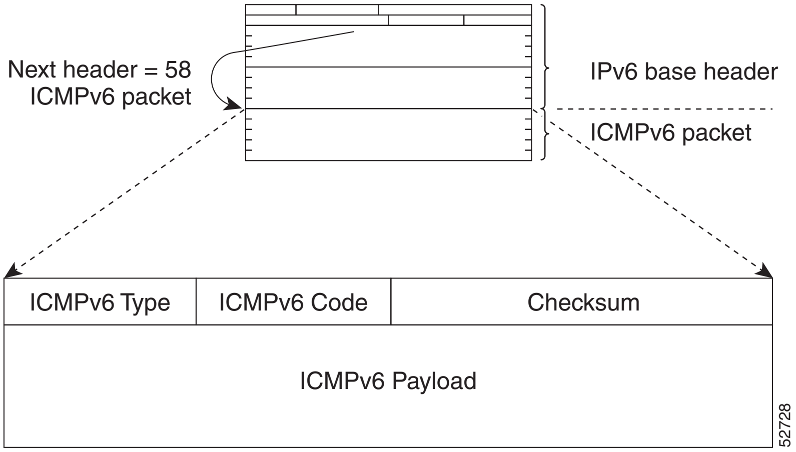
You can use ICMP in IPv6 to provide information about the health of the network. ICMPv6, the version that works with IPv6, reports errors if packets cannot be processed correctly and sends informational messages about the status of the network. For example, if a router cannot forward a packet because it is too large to be sent out on another network, the router sends out an ICMPv6 message to the originating host. Additionally, ICMP packets in IPv6 are used in IPv6 neighbor discovery and path MTU discovery. The path MTU discovery process ensures that a packet is sent using the largest possible size that is supported on a specific route.

A value of 58 in the Next Header field of the base IPv6 packet header identifies an IPv6 ICMP packet. The ICMP packet follows all the extension headers and is the last piece of information in the IPv6 packet.Within the IPv6 ICMP packets, the ICMPv6 Type and ICMPv6 Code fields identify IPv6 ICMP packet specifics, such as the ICMP message type. The value in the Checksum field is computed by the sender and checked by the receiver from the fields in the IPv6 ICMP packet and the IPv6 pseudo header.

Note The IPv6 header does not have a checksum. But a checksum on the transport layer can determine if packets have not been delivered correctly. All checksum calculations that include the IP address in the calculation must be modified for IPv6 to accommodate the new 128-bit address. A checksum is generated using a pseudo header.

The ICMPv6 Payload field contains error or diagnostic information that relates to IP packet processing. Figure 1-11 shows the IPv6 ICMP packet header format.

Figure 1-11 IPv6 ICMP Packet Header Format



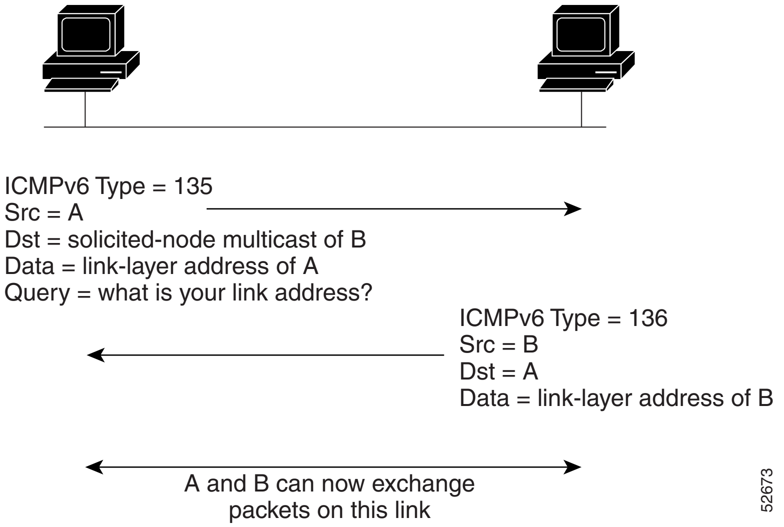
IPv6 Neighbor Discovery

You can use the IPv6 Neighbor Discovery Protocol (NDP) to determine whether a neighboring router is reachable. IPv6 nodes use neighbor discovery to determine the addresses of nodes on the same network (local link), to find neighboring routers that can forward their packets, to verify whether neighboring routers are reachable or not, and to detect changes to link-layer addresses. NDP uses ICMP messages to detect whether packets are sent to neighboring routers that are unreachable.

IPv6 Neighbor Solicitation Message

A node sends a neighbor solicitation message, which has a value of 135 in the Type field of the ICMP packet header, on the local link when it wants to determine the link-layer address of another node on the same local link (see Figure 1-12). The source address is the IPv6 address of the node that sends the neighbor solicitation message. The destination address is the solicited-node multicast address that corresponds to the IPv6 address of the destination node. The neighbor solicitation message also includes the link-layer address of the source node.

Figure 1-12 IPv6 Neighbor Discovery—Neighbor Solicitation Message


ICMPv6 Type = 135
Src = A
Dst = solicited-node multicast of B
Data = link-layer address of A
Query = what is your link address?
ICMPv6 Type = 136
Src = B
Dst = A
Data = link-layer address of B
A and B can now exchange packets on this link
52673

After receiving the neighbor solicitation message, the destination node replies by sending a neighbor advertisement message, which has a value of 136 in the Type field of the ICMP packet header, on the local link. The source address is the IPv6 address of the node (the IPv6 address of the node interface that sends the neighbor advertisement message). The destination address is the IPv6 address of the node that sends the neighbor solicitation message. The data portion includes the link-layer address of the node that sends the neighbor advertisement message.

After the source node receives the neighbor advertisement, the source node and destination node can communicate.

Neighbor solicitation messages can verify the reachability of a neighbor after a node identifies the link-layer address of a neighbor. When a node wants to verify the reachability of a neighbor, it uses the destination address in a neighbor solicitation message as the unicast address of the neighbor.

Neighbor advertisement messages are also sent when there is a change in the link-layer address of a node on a local link. When there is a change, the destination address for the neighbor advertisement is the all-nodes multicast address.

Neighbor unreachability detection identifies the failure of a neighbor or the failure of the forward path to the neighbor and is used for all paths between hosts and neighboring nodes (hosts or routers). Neighbor unreachability detection is performed for neighbors to which only unicast packets are being sent and is not performed for neighbors to which multicast packets are being sent.

A neighbor is considered reachable when a positive acknowledgment is returned from the neighbor (indicating that packets previously sent to the neighbor have been received and processed). A positive acknowledgment—from an upper-layer protocol (such as TCP)—indicates that a connection is making forward progress (reaching its destination). If packets are reaching the peer, they are also reaching the next-hop neighbor of the source. Forward progress is also a confirmation that the next-hop neighbor is reachable.

For destinations that are not on the local link, forward progress implies that the first-hop router is reachable. When acknowledgments from an upper-layer protocol are not available, a node probes the neighbor using unicast neighbor solicitation messages to verify that the forward path is still working. The return of a solicited neighbor advertisement message from the neighbor is a positive acknowledgment that the forward path is still working (neighbor advertisement messages that have the solicited flag set to a value of 1 are sent only in response to a neighbor solicitation message). Unsolicited messages confirm only the one-way path from the source to the destination node; solicited neighbor advertisement messages indicate that a path is working in both directions.

Note A neighbor advertisement message that has the solicited flag set to a value of 0 is not considered as a positive acknowledgment that the forward path is still working.

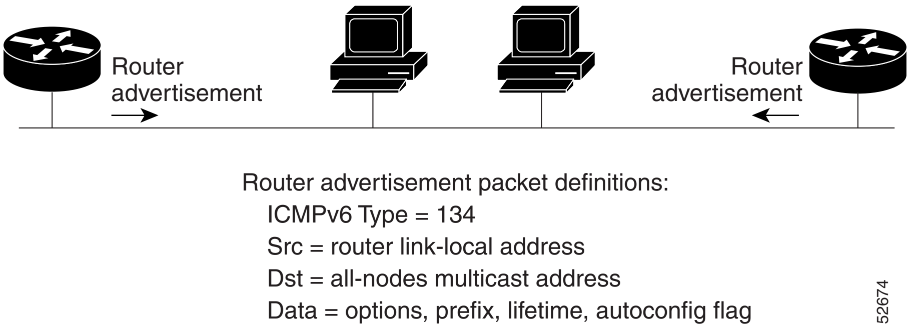
Neighbor solicitation messages are also used in the stateless autoconfiguration process to verify the uniqueness of unicast IPv6 addresses before the addresses are assigned to an interface. Duplicate address detection is performed first on a new, link-local IPv6 address before the address is assigned to an interface (the new address remains in a tentative state while duplicate address detection is performed). A node sends a neighbor solicitation message with an unspecified source address and a tentative link-local address in the body of the message. If another node is already using that address, the node returns a neighbor advertisement message that contains the tentative link-local address. If another node is simultaneously verifying the uniqueness of the same address, that node also returns a neighbor solicitation message. If no neighbor advertisement messages are received in response to the neighbor solicitation message and no neighbor solicitation messages are received from other nodes that are attempting to verify the same tentative address, the node that sent the original neighbor solicitation message considers the tentative link-local address to be unique and assigns the address to the interface.

IPv6 Router Advertisement Message

Router advertisement (RA) messages, which have a value of 134 in the Type field of the ICMP packet header, are periodically sent out to each configured interface of an IPv6 router. For stateless autoconfiguration to work properly, the advertised prefix length in RA messages must always be 64 bits.

The RA messages are sent to the all-nodes multicast address (see Figure 1-13).

Figure 1-13 IPv6 Neighbor Discovery—RA Message



Router advertisement packet definitions:
ICMPv6 Type = 134
Src = router link-local address
Dst = all-nodes multicast address
Data = options, prefix, lifetime, autoconfig flag

RA messages typically include the following information:

- One or more onlink IPv6 prefixes that nodes on the local link can use to automatically configure their IPv6 addresses
- Life-time information for each prefix included in the advertisement
- Sets of flags that indicate the type of autoconfiguration (stateless or stateful) that can be completed
- Default router information (whether the router sending the advertisement should be used as a default router and, if so, the amount of time in seconds that the router should be used as a default router)
- Additional information for hosts, such as the hop limit and MTU that a host should use in packets that it originates

RAs are also sent in response to router solicitation messages. Router solicitation messages, which have a value of 133 in the Type field of the ICMP packet header, are sent by hosts at system startup so that the host can immediately autoconfigure without needing to wait for the next scheduled RA message. The source address is usually the unspecified IPv6 address (0:0:0:0:0:0:0:0). If the host has a configured unicast address, the unicast address of the interface that sends the router solicitation message is used as the source address in the message. The destination address is the all-routers multicast address with a scope of the link. When an RA is sent in response to a router solicitation, the destination address in the RA message is the unicast address of the source of the router solicitation message.

You can configure the following RA message parameters:

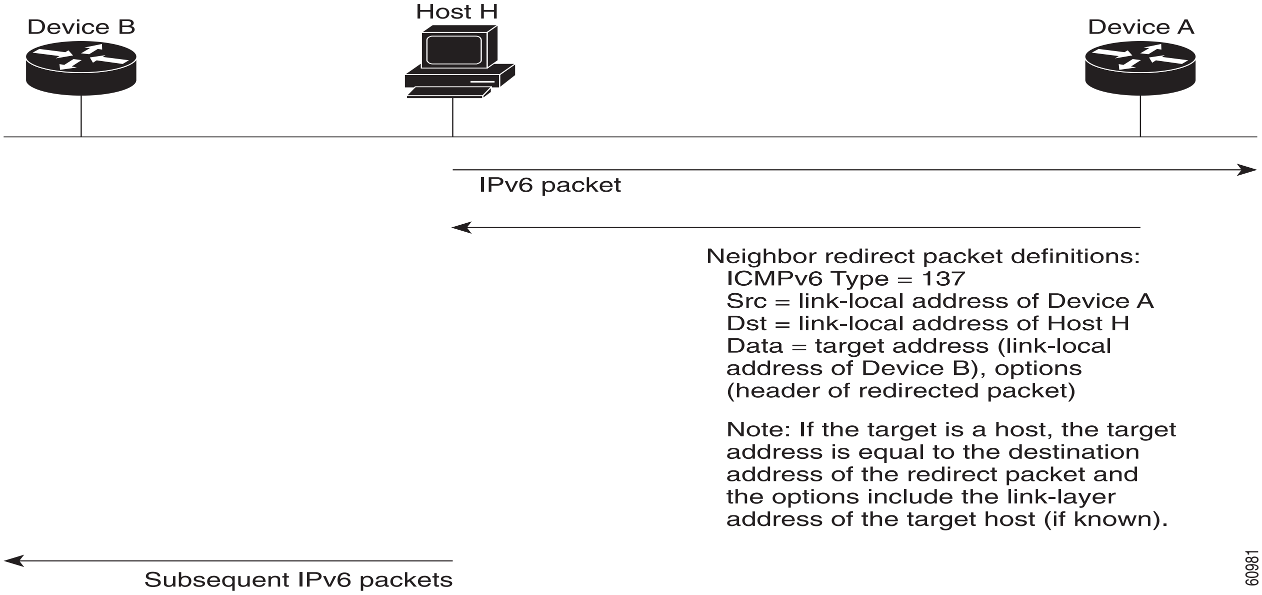
- The time interval between periodic RA messages
- The router life-time value, which indicates the usefulness of a router as the default router (for use by all nodes on a given link)
- The network prefixes in use on a given link
- The time interval between neighbor solicitation message retransmissions (on a given link)
- The amount of time that a node considers a neighbor reachable (for use by all nodes on a given link)

The configured parameters are specific to an interface. The sending of RA messages (with default values) is automatically enabled on Ethernet interfaces. For other interface types, you must enter the **no ipv6 nd suppress-ra** command to send RA messages. You can disable the RA message feature on individual interfaces by entering the **ipv6 nd suppress-ra** command.

IPv6 Neighbor Redirect Message

Routers send neighbor redirect messages to inform hosts of better first-hop nodes on the path to a destination (see Figure 1-14). A value of 137 in the Type field of the ICMP packet header identifies an IPv6 neighbor redirect message.

Figure 1-14 IPv6 Neighbor Discovery—Neighbor Redirect Message


Device B
Host H
Device A
IPv6 packet
Neighbor redirect packet definitions:
ICMPv6 Type = 137
Src = link-local address of Device A
Dst = link-local address of Host H
Data = target address (link-local address of Device B), options (header of redirected packet)
Note: If the target is a host, the target address is equal to the destination address of the redirect packet and the options include the link-layer address of the target host (if known).
Subsequent IPv6 packets
60981

Note A router must be able to determine the link-local address for each of its neighboring routers in order to ensure that the target address (the final destination) in a redirect message identifies the neighbor router by its link-local address. For static routing, you should specify the address of the next-hop router using the link-local address of the router. For dynamic routing, you must configure all IPv6 routing protocols to exchange the link-local addresses of neighboring routers.

After forwarding a packet, a router sends a redirect message to the source of the packet under the following circumstances:

- The destination address of the packet is not a multicast address.
- The packet was not addressed to the router.
- The packet is about to be sent out the interface on which it was received.
- The router determines that a better first-hop node for the packet resides on the same link as the source of the packet.
- The source address of the packet is a global IPv6 address of a neighbor on the same link or a link-local address.

Virtualization Support

IPv6 supports virtual routing and forwarding (VRF) instances.

Licensing Requirements for IPv6

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	IPv6 requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for IPv6

IPv6 has the following prerequisites:

- You must be familiar with IPv6 basics such as IPv6 addressing, IPv6 header information, ICMPv6, and the IPv6 Neighbor Discovery (ND) Protocol.
- Ensure that you follow the memory/processing guidelines when you make a device a dual-stack device (IPv4/IPv6).

Guidelines and Limitations for IPv6

IPv6 has the following configuration guidelines and limitations:

- IPv6 packets are transparent to Layer 2 LAN switches because the switches do not examine Layer 3 packet information before forwarding IPv6 frames. IPv6 hosts can be directly attached to Layer 2 LAN switches.
- You can configure multiple IPv6 global addresses within the same prefix on an interface. However, multiple IPv6 link-local addresses on an interface are not supported.
- Because RFC 3879 deprecates the use of site-local addresses, you should configure private IPv6 addresses according to the recommendations of unique local addressing (ULA) in RFC 4193.

Default Settings

Table 1-4 lists the default settings for IPv6 parameters.

Table 1-4 Default IPv6 Parameters

Parameters	Default
ND reachable time	0 milliseconds
neighbor solicitation retransmit interval	1000 milliseconds

Configuring IPv6

This section includes the following topics:

- Configuring IPv6 Addressing, page 1-19
- Configuring IPv6 Neighbor Discovery, page 1-21
- Optional IPv6 Neighbor Discovery, page 1-23

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 IPv6 Addressing

You must configure an IPv6 address on an interface so that the interface can forward IPv6 traffic. When you configure a global IPv6 address on an interface, it automatically configures a link-local address and activates IPv6 for that interface

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **ipv6 address** {*addr* [**eui64**] [**route-preference** *preference*] [**secondary**] tag *tag-id*]*]*

or

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

4. **(Optional) show ipv6 interface**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface ethernet`** *`number`*
	Example:
	`switch(config)# interface ethernet 2/3`
	`switch(config-if)#`
		Enters interface configuration mode.

	Command	Purpose

Step 3	**ipv6 address** {*addr* [**eui64**]
	[**route-preference** *preference*]
	[**secondary**] **tag** *tag-id*]
	or
	ipv6 address *ipv6-address*
	use-link-local-only
	
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/10`
	or
	`switch(config-if)# ipv6 address use-link-local-only`
		Specifies an IPv6 address assigned to the interface and enables IPv6 processing on the interface.
		Entering the **ipv6 address** command configures global IPv6 addresses with an interface identifier (ID) in the low-order 64 bits of the IPv6 address. Only the 64-bit network prefix for the address needs to be specified; the last 64 bits are automatically computed from the interface ID.
		Entering the **ipv6 address use-link-local-only command configures a link-local address on the interface that is used instead of the link-local address that is automatically configured when IPv6 is enabled on the interface.**
		This command enables IPv6 processing on an interface without configuring an IPv6 address.

Step 4	**show ipv6 interface** **Example:** `switch(config-if)# show ipv6 interface`	(Optional) Displays interfaces configured for IPv6.
Step 5	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure an IPv6 address:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# ipv6 address ?
A:B::C:D/LEN IPv6 prefix format: xxxx:xxxx/ml, xxxx:xxxx::/ml,
xxxx::xx/128
use-link-local-only  Enable IPv6 on interface using only a single link-local
address
switch(config-if)# ipv6 address 2001:db8::/64 eui64
```

This example shows how to display an IPv6 interface:

```
switch(config-if)# show ipv6 interface ethernet 3/1
Ethernet3/1, Interface status: protocol-down/link-down/admin-down, iod: 36
    IPv6 address: 0dc3:0dc3:0000:0000:0218:baff:fed8:239d
    IPv6 subnet:  0dc3:0dc3:0000:0000:0000:0000:0000:0000/64
    IPv6 link-local address: fe80::0218:baff:fed8:239d (default)
    IPv6 multicast routing: disabled
    IPv6 multicast groups locally joined:
        ff02::0001:ffd8:239d  ff02::0002  ff02::0001  ff02::0001:ffd8:239d
    IPv6 multicast (S,G) entries joined: none
    IPv6 MTU: 1500 (using link MTU)
    IPv6 RP inbound packet-filtering policy: none
    IPv6 RP outbound packet-filtering policy: none
    IPv6 inbound packet-filtering policy: none
    IPv6 outbound packet-filtering policy: none
    IPv6 interface statistics last reset: never
    IPv6 interface RP-traffic statistics: (forwarded/originated/consumed)
        Unicast packets: 0/0/0
        Unicast bytes: 0/0/0
```

```
Multicast packets: 0/0/0
Multicast bytes: 0/0/0
```

Configuring IPv6 Neighbor Discovery

You can configure IPv6 neighbor discovery on the router. NDP enables IPv6 nodes and routers to determine the link-layer address of a neighbor on the same link, find neighboring routers, and keep track of neighbors.

SUMMARY STEPS

1. **configure terminal**
2. **interface ethernet** *number*
3. **ipv6 nd [hop-limit** *hop-limit* **| managed-config-flag | mtu** *mtu* **| ns-interval** *interval* **| other-config-flag | prefix | ra-interval** *interval* **| ra-lifetime** *lifetime* **| reachable-time** *time* **| redirects | retrans-timer** *time* **| suppress-ra]**
4. **(Optional) show ipv6 nd interface**
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface ethernet`** *`number`*
	
	`Example:`
	`switch(config)# interface ethernet 2/31`
	`switch(config-if)#`
		Enters interface configuration mode.

	Command	Purpose
Step 3	**ipv6 nd [hop-limit** *hop-limit* **| managed-config-flag | mtu** *mtu* **| ns-interval** *interval* **| other-config-flag | prefix | ra-interval** *interval* **| ra-lifetime** *lifetime* **| reachable-time** *time* **| redirects | retrans-timer** *time* **| suppress-ra]** **Example:** `switch(config-if)# ipv6 nd prefix`	Neighbor discovery is enabled automatically when you configure an IPv6 address. This command enables the following additional IPv6 neighbor discovery options on the interface: • **hop-limit** *hop-limit*—Advertises the hop limit in IPv6 neighbor discovery packets. The range is from 0 to 255. • **managed-config-flag**—Advertises in ICMPv6 router-advertisement messages to use stateful address autoconfiguration to obtain address information. • **mtu** *mtu*—Advertises the maximum transmission unit (MTU) in ICMPv6 router-advertisement messages on this link. The range is from 1280 to 65535 bytes. • **ns-interval** *interval*—Configures the retransmission interval between IPv6 neighbor solicitation messages. The range is from 1000 to 3600000 milliseconds. • **other-config-flag**—Indicates in ICMPv6 router-advertisement messages that hosts use stateful auto configuration to obtain nonaddress related information. • **prefix**—Advertises the IPv6 prefix in the router-advertisement messages. • **ra-interval** *interval*—Configures the interval between sending ICMPv6 router-advertisement messages. The range is from 4 to 1800 seconds. • **ra-lifetime** *lifetime*—Advertises the lifetime of a default router in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds. • **reachable-time** *time*—Advertises the time when a node considers a neighbor up after receiving a reachability confirmation in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds. • **redirects**—Enables sending ICMPv6 redirect messages. • **retrans-timer** *time*—Advertises the time between neighbor-solicitation messages in ICMPv6 router-advertisement messages. The range is from 0 to 9000 seconds. • **suppress-ra**—Disables sending ICMPv6 router-advertisement messages.

	Command	Purpose
Step 4	**show ipv6 nd interface** **Example:** switch(config-if)# show ipv6 nd interface	(Optional) Displays interfaces configured for IPv6 neighbor discovery.
Step 5	**copy running-config startup-config** **Example:** switch(config-if)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure IPv6 neighbor discovery reachable time:

```
switch# configure terminal
switch(config)# interface ethernet 3/1
switch(config-if)# ipv6 nd reachable-time 10
```

This example shows how to display an IPv6 neighbor discovery interface:

```
switch(config-if)# show ipv6 nd interface ethernet 3/1
ICMPv6 ND Interfaces for VRF "default"
Ethernet3/1, Interface status: protocol-down/link-down/admin-down
IPv6 address: 0dc3:0dc3:0000:0000:0218:baff:fed8:239d
    ICMPv6 active timers:
        Last Neighbor-Solicitation sent: never
        Last Neighbor-Advertisement sent: never
        Last Router-Advertisement sent:never
        Next Router-Advertisement sent in: 0.000000
    Router-Advertisement parameters:
        Periodic interval: 200 to 600 seconds
        Send "Managed Address Configuration" flag: false
        Send "Other Stateful Configuration" flag: false
        Send "Current Hop Limit" field: 64
        Send "MTU" option value: 1500
        Send "Router Lifetime" field: 1800 secs
        Send "Reachable Time" field: 10 ms
        Send "Retrans Timer" field: 0 ms
    Neighbor-Solicitation parameters:
        NS retransmit interval: 1000 ms
    ICMPv6 error message parameters:
        Send redirects: false
        Send unreachables: false
```

Optional IPv6 Neighbor Discovery

You can use the following optional IPv6 Neighbor Discovery commands:

Command	Purpose
ipv6 nd hop-limit	Configures the maximum number of hops used in router advertisements and all IPv6 packets that are originated by the router.
ipv6 nd managed-config-flag	Sets the managed address configuration flag in IPv6 router advertisements.

Command	Purpose
ipv6 nd mtu	Sets the maximum transmission unit (MTU) size of IPv6 packets sent on an interface.
ipv6 nd ns-interval	Configures the interval between IPv6 neighbor solicitation retransmissions on an interface.
ipv6 nd other-config-flag	Configures the other stateful configuration flag in IPv6 router advertisements.
ipv6 nd ra-interval	Configures the interval between IPv6 router advertisement (RA) transmissions on an interface.
ipv6 nd ra-lifetime	Configures the router lifetime value in IPv6 router advertisements on an interface.
ipv6 nd reachable-time	Configures the amount of time that a remote IPv6 node is considered reachable after some reachability confirmation event has occurred.
ipv6 nd redirects	Enables ICMPv6 redirect messages to be sent.
ipv6 nd retrans-timer	Configures the advertised time between neighbor solicitation messages in router advertisements.
ipv6 nd suppress-ra	Suppresses IPv6 router advertisement transmissions on a LAN interface.

Verifying the IPv6 Configuration

To display the IPv6 configuration, perform one of the following tasks:

Command	Purpose
show ipv6 interface	Displays IPv6-related interface information.
show ipv6 adjacency	Displays the adjacency table.
show ipv6 icmp	Displays ICMPv6 information.
show ipv6 nd	Displays IPv6 neighbor discovery interface information.
show ipv6 neighbor	Displays IPv6 neighbor entry.

Configuration Examples for IPv6

This example shows how to configure IPv6:

```
configure terminal
 interface ethernet 3/1
  ipv6 address 2001:db8::/64 eui64
   ipv6 nd reachable-time 10
```

Additional References

For additional information related to implementing IPv6, see the following sections:

- Related Documents, page 1-25
- Standards, page 1-25

Related Documents

Related Topic	Document Title
IPv6 CLI commands	*Cisco Nexus 6000 Series NX-OS Unicast Routing Command Reference, Release 6.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

Configuring OSPFv2

This chapter describes how to configure Open Shortest Path First version 2 (OSPFv2) for IPv4 networks.

This chapter includes the following sections:

Information About OSPFv2

OSPFv2 is an IETF link-state protocol (see the "Link-State Protocols" section on page 1-9) 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 attempt 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* (see the "Convergence" section on page 1-6). 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, while OSPFv3 supports IPv6. For more information, see Chapter 1, "Configuring OSPFv3."

This section includes the following topics:

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 the "Designated Routers" section on page 1-3)

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" section on page 1-2).

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, then 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" section on page 1-4)
- 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" section on page 1-3).
- 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" section on page 1-3).
- 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" section on page 1-3.

Adjacency is established using Database Description packets, Link State Request packets, and Link State Update packets in OSPF. The Database Description packet includes only the LSA headers from the link-state database of the neighbor (see the "Link-State Database" section on page 1-7). The local router compares these headers with its own link-state database and determines which LSAs are new or updated. The local router sends a Link State Request packet for each LSA that it needs new or updated information on. The neighbor responds with a Link State Update 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 will be 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 on page 1-4). If the DR fails, OSPFv2 selects a *backup designated router* (BDR). If the DR fails, OSPFv2 uses the BDR.

Network types are as follows:

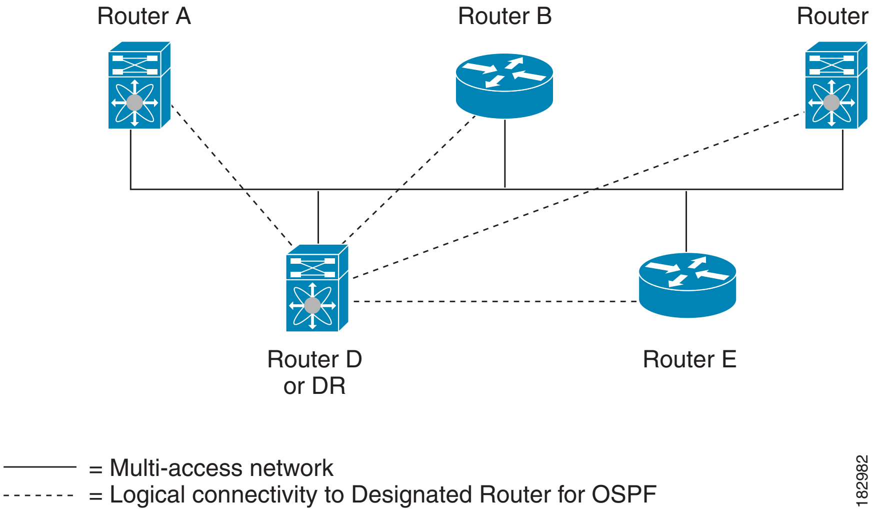
- Point-to-point—A network that exists only between two routers. All neighbors on a point-to-point network establish adjacency and there is no DR.
- Broadcast—A network with multiple routers that can communicate over a shared medium that allows broadcast traffic, such as Ethernet. OSPFv2 routers establish a DR and 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. Figure 1-1 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 1-1 DR in Multi-Access Network


Router A
Router B
Router C
Router D
or DR
Router E
——— = Multi-access network
------- = Logical connectivity to Designated Router for OSPF
182982

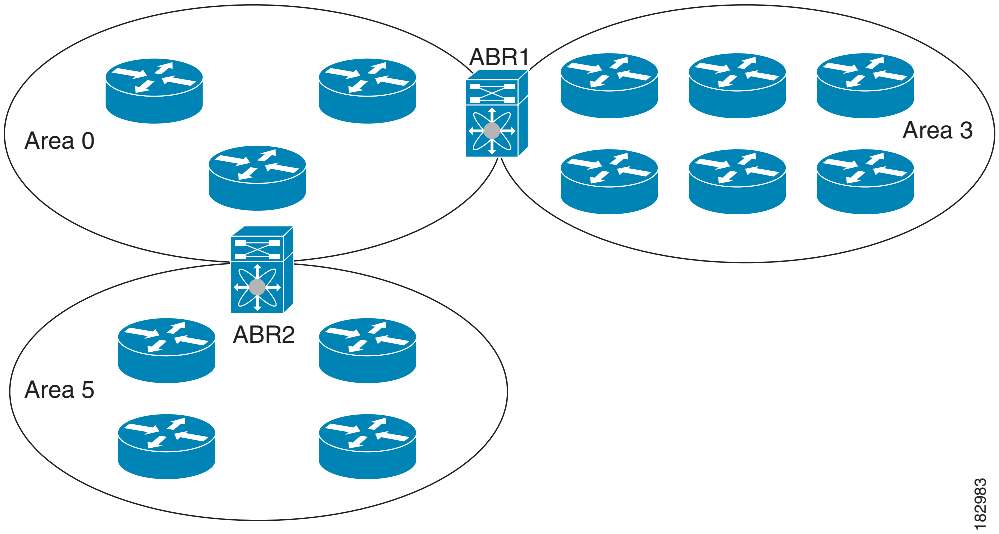
Areas

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). An ABR connects to both the backbone area and at least one other defined area (see Figure 1-2).

Figure 1-2 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" section on page 1-10) from one connected area to the backbone area. The backbone area sends summarized information about one area to another area. In Figure 1-2, 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 "Advanced Features" section on page 1-8.)

Link-State Advertisements

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

This section includes the following topics:

- LSA Types, page 1-5
- Link Cost, page 1-6
- Flooding and LSA Group Pacing, page 1-6
- Link-State Database, page 1-7
- Opaque LSAs, page 1-7

LSA Types

Table 1-1 shows the LSA types supported by Cisco NX-OS.

Table 1-1 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 on page 1-3.
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 on page 1-4.
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 on page 1-4.
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 on page 1-4.
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 on page 1-4.
9–11	Opaque LSAs	LSA used to extend OSPF. See the "Opaque LSAs" section on page 1-7.

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" section on page 1-4). 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 utilization. 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 four minutes 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" section on page 1-6.

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. 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 on page 1-11)

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.

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.

Advanced Features

Cisco NX-OS supports a number of advanced OSPFv2 features that enhance the usability and scalability of OSPFv2 in the network. This section includes the following topics:

- Stub Area, page 1-8
- Not-So-Stubby Area, page 1-9
- Virtual Links, page 1-9
- Route Redistribution, page 1-10
- Route Summarization, page 1-10OSPFv2 Stub Router Advertisements, page 1-11
- Multiple OSPFv2 Instances, page 1-11
- SPF Optimization, page 1-11
- BFD, page 1-11
- Virtualization Support, page 1-12

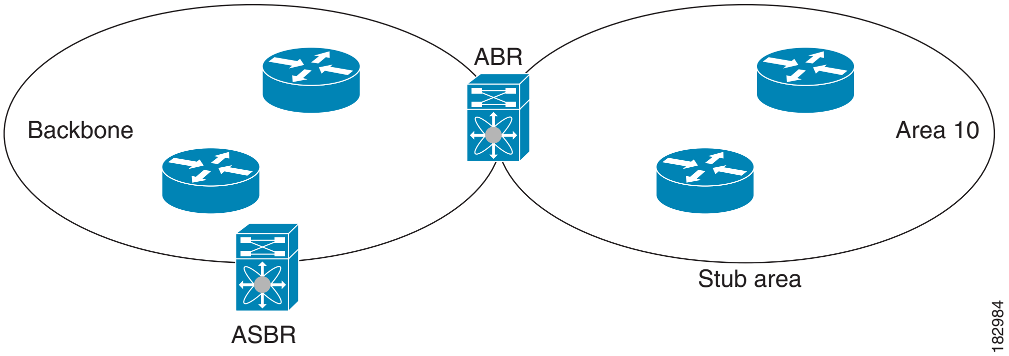
Stub Area

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 Advertisements" section on page 1-5). 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. See the "Stub Routing" section on page 1-7.
- No ASBR routers exist in the stub area.
- You cannot configure virtual links in the stub area.

Figure 1-3 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 1-3 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 area border router (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" section on page 1-5 for details on NSSA External LSAs.

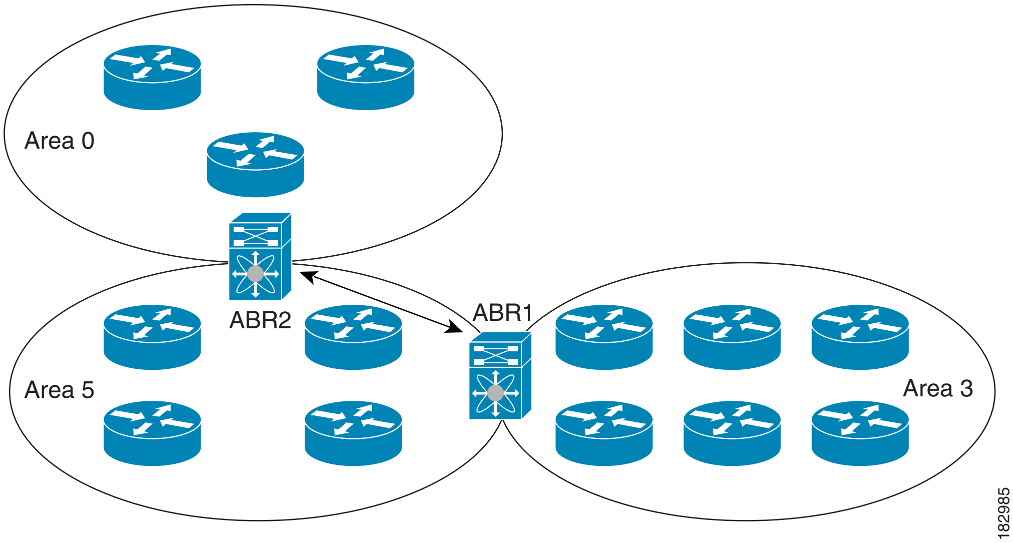
You can, for example, use NSSA to simplify administration if you are connecting a central site using OSPFv2 to a remote site that is using a different routing protocol. Before NSSA, the connection between the corporate site border router and a remote router could not be run as an OSPFv2 stub area because 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 (see the "Configuring NSSA" section on page 1-26).

The backbone Area 0 cannot be an NSSA.

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. Figure 1-4 shows a virtual link that connects Area 3 to the backbone area through Area 5.

Figure 1-4 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. See the "Route Redistribution" section on page 1-6. 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. See Chapter 1, "Configuring Route Policy Manager," for details on configuring route maps. 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.

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.

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 NX-OS 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.

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.

BFD

OSPFv2 supports bidirectional forwarding detection (BFD). BFD is a detection protocol that provides fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules. See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x* for more information.

Virtualization Support

OSPFv2 supports Virtual Routing and Forwarding (VRFs) instances. Each OSPFv2 instance can support multiple VRFs, up to the system limit.

Licensing Requirements for OSPFv2

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	OSPFv2 requires a LAN Base Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for 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 is capable of communicating with a remote OSPFv2 neighbor.
- You have installed the LAN Base Services license.
- 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 the OSPFv2 Feature" section on page 1-13).

Guidelines and Limitations

OSPFv2 has the following configuration guidelines and limitations:

- You can have up to four instances of OSPFv2.
- You can have up to four instances of OSPFv2 in a VDC.
- Cisco NX-OS displays areas in dotted decimal notation regardless of whether you enter the area in decimal or dotted decimal notation.
- If you configure OSPF in a vPC environment, use the following timer commands in router configuration mode on the core switch to ensure fast OSPF convergence when a vPC peer-link is shut down:

```
switch(config-router)# timers throttle spf 1 50 50
switch(config-router)# timers lsa-arrival 10
```

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

Table 1-2 lists the default settings for OSPFv2 parameters.

Table 1-2 Default OSPFv2 Parameters

Parameters	Default
Hello interval	10 seconds
Dead interval	40 seconds
Graceful restart notify period	15 seconds
OSPFv2 feature	Disabled
Stub router advertisement announce time	600 seconds
Reference bandwidth for link cost calculation	40 Gb/s
LSA minimal arrival time	1000 milliseconds
LSA group pacing	240 seconds
SPF calculation initial delay time	0 milliseconds
SPF calculation hold time	5000 milliseconds
SPF calculation initial delay time	0 milliseconds

Configuring Basic OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.This section includes the following topics:

- Enabling the OSPFv2 Feature, page 1-13
- Creating an OSPFv2 Instance, page 1-14
- Configuring Optional Parameters on an OSPFv2 Instance, page 1-15
- Configuring Optional Parameters on an OSPFv2 Instance, page 1-15
- Configuring Networks in OSPFv2, page 1-16
- Configuring Authentication for an Area, page 1-18
- Configuring Authentication for an Interface, page 1-20

Enabling the OSPFv2 Feature

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

SUMMARY STEPS

1. **configure terminal**
2. **feature ospf**
3. (Optional) **show feature**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`feature ospf`**
	Example:
	`switch(config)# feature ospf`
		Enables the OSPFv2 feature.

Step 3	**`show feature`**
	Example:
	`switch(config)# show feature`
		(Optional) Displays enabled and disabled features.

Step 4	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

Use the **no feature ospf** command to disable the OSPFv2 feature and remove all associated configurations.

Command	Purpose

`no feature ospf`
Example:
`switch(config)# no feature ospf`
	Disables the OSPFv2 feature and removes all associated configurations.

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" section on page 1-22.

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

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.

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*

3. (Optional) **router-id** *ip-address*
4. (Optional) **show ip ospf** *instance-tag*
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router ospf** *instance-tag* **Example:** switch(config)# router ospf 201 switch(config-router)#	Creates a new OSPFv2 instance with the configured instance tag.
Step 3	**router-id** *ip-address* **Example:** switch(config-router)# router-id 192.0.2.1	(Optional) Configures the OSPFv2 router ID. This IP address identifies this OSPFv2 instance and must exist on a configured interface in the system.
Step 4	**show ip ospf** *instance-tag* **Example**: switch(config-router)# show ip ospf 201	(Optional) Displays OSPF information.
Step 5	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **no router ospf** command to remove the OSPFv2 instance and all associated configurations.

Command	Purpose
no router ospf *instance-tag* **Example:** switch(config)# no router ospf 201	Deletes the OSPF instance and the associated configurations.

This command does not remove 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.

For more information about OSPFv2 instance parameters, see the "Configuring Advanced OSPFv2" section on page 1-22.

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

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

DETAILED STEPs

Command	Purpose
distance *number* **Example:** `switch(config-router)# distance 25`	Configures the administrative distance for this OSPFv2 instance. The range is from 1 to 255. The default is 110.
log-adjacency-changes [**detail**] **Example:** `switch(config-router)# log-adjacency-changes`	Generates a system message whenever a neighbor changes state.
maximum-paths *path-number* **Example:** `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 64. The default is 8.

This example shows how to create an OSPFv2 instance with a maximum of four equal paths per destination:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# maximum-paths 4
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 on page 1-2). 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 the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **ip address** *ip-prefix/length*
5. **ip router ospf** *instance-tag* **area** *area-id* [**secondaries none**]
6. (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface** *interface-type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	no switchport **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 4	**ip address** *ip-prefix/length* **Example:** `switch(config-if)# ip address 192.0.2.1/16`	Assigns an IP address and subnet mask to this interface.
Step 5	**ip router ospf** *instance-tag* **area** *area-id* [**secondaries none**] **Example:** `switch(config-if)# ip router ospf 201 area 0.0.0.15`	Adds the interface to the OSPFv2 instance and area.
Step 6	**show ip ospf** *instance-tag* **interface** *interface-type slot/port* **Example**: `switch(config-if)# show ip ospf 201 interface ethernet 1/2`	(Optional) Displays OSPF information. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

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

Command	Purpose
ip ospf cost *number* **Example:** `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.
ip ospf dead-interval *seconds* **Example:** `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.
ip ospf hello-interval *seconds* **Example:** `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.
ip ospf mtu-ignore **Example:** `switch(config-if)# ip ospf mtu-ignore`	Configures OSPFv2 to ignore any IP MTU mismatch with a neighbor. The default is to not establish adjacency if the neighbor MTU does not match the local interface MTU.
ip ospf passive-interface **Example:** `switch(config-if)# ip ospf passive-interface`	Suppresses routing updates on the interface.
ip ospf priority *number* **Example:** `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 on page 1-3.
ip ospf shutdown **Example:** `switch(config-if)# ip ospf shutdown`	Shuts down the OSPFv2 instance on this interface.

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)# no switchport
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 the OSPFv2 Feature" section on page 1-13).

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 6000 Series NX-OS Security Configuration Guide, Release 6.0.*

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **area** *area-id* **authentication** [**message-digest**]
4. **interface** *interface-type slot/port*
5. **no switchport**
6. (Optional) **ip ospf authentication-key** [**0** | **3**] *key*
   or
   (Optional) **ip ospf message-digest-key** *key-id* **md5** [**0** | **3**] *key*
7. (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **authentication** [**message-digest**] **Example:** `switch(config-router)# area 0.0.0.10 authentication`	Configures the authentication mode for an area.

Step 4	**interface** *interface-type slot/port*
	Example:
	`switch(config-router)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

	Command	Purpose
Step 5	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 6	**ip ospf authentication-key** [**0** | **3**] *key* **Example:** `switch(config-if)# ip ospf authentication-key 0 mypass`	(Optional) 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.
	ip ospf message-digest-key *key-id* **md5** [**0** | **3**] *key* **Example:** `switch(config-if)# ip ospf message-digest-key 21 md5 0 mypass`	(Optional) 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	**show ip ospf** *instance-tag* **interface** *interface-type slot/port* **Example:** `switch(config-if)# show ip ospf 201 interface ethernet 1/2`	(Optional) Displays OSPF information. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 8	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring Authentication for an Interface

You can configure authentication 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 the OSPFv2 Feature" section on page 1-13).

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 6000 Series NX-OS Security Configuration Guide, Release 6.0.*

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **ip ospf authentication** [**message-digest**]
5. *(Optional)* **ip ospf authentication key-chain** *key-id*

6. (Optional) **ip ospf authentication-key** [**0** | **3**] *key*
7. (Optional) **ip ospf message-digest-key** *key-id* **md5** [**0** | **3**] *key*
8. (Optional) **show ip ospf** *instance-tag* **interface** *interface-type slot/port*
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**interface** *interface-type slot/port* **Example:** switch(config)# interface ethernet 1/2 switch(config-if)#	Enters interface configuration mode. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 3	**no switchport** **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 4	**ip ospf authentication** [**message-digest**] **Example:** 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 5	**ip ospf authentication key-chain** *key-name* **Example:** switch(config-if)# ip ospf authentication key-chain Test1	(Optional) Configures interface authentication to use key chains for OSPFv2. See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*, for details on key chains.
Step 6	**ip ospf authentication-key** [**0** | **3** | **7**] *key* **Example:** switch(config-if)# ip ospf authentication-key 0 mypass	(Optional) Configures simple password authentication for this interface. Use this command if the authentication is not set to key-chain or message-digest. The options are as follows: • 0—configures the password in clear text. • 3—configures the pass key as 3DES encrypted. • 7—configures the key as Cisco type 7 encrypted.

	Command	Purpose
Step 7	**ip ospf message-digest-key** *key-id* **md5** [**0** | **3** | **7**] *key* **Example:** `switch(config-if)# ip ospf message-digest-key 21 md5 0 mypass`	(Optional) 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: • 0—configures the password in clear text. • 3—configures the pass key as 3DES encrypted. • 7—configures the key as Cisco type 7 encrypted.
Step 8	**show ip ospf** *instance-tag* **interface** *interface-type slot/port* **Example:** `switch(config-if)# show ip ospf 201 interface ethernet 1/2`	(Optional) Displays OSPF information. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 9	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

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)# no switchport
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
```

Configuring Advanced OSPFv2

Configure OSPFv2 after you have designed your OSPFv2 network.

This section includes the following topics:

- Modifying the Default Timers, page 1-36
- Restarting an OSPFv2 Instance, page 1-39

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 also can connect to external domains, through an *autonomous system border router* (ASBR). See the “Areas” section on page 1-4.

ABRs have the following optional configuration parameters:

- Area range—Configures route summarization between areas.
- Filter list—Filters the Network Summary (type 3) LSAs on an ABR 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 the OSPFv2 Feature” section on page 1-13).

Create the route map that the filter list uses to filter IP prefixes in incoming or outgoing Network Summary (type 3) LSAs. See Chapter 1, “Configuring Route Policy Manager.”

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **area** *area-id* **filter-list route-map** *map-name* {**in** | **out**}
4. (Optional) **show ip ospf policy statistics**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

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

	Command	Purpose

Step 3	**area** *area-id* **filter-list route-map** *map-name* {**in** | **out**}
	Example:
	`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	**show ip ospf policy statistics area** *id* **filter-list** {**in** | **out**} **Example:** `switch(config-if)# show ip ospf policy statistics area 0.0.0.10 filter-list in`	(Optional) Displays OSPF policy information.
Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

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, limiting unnecessary routing to and from selected networks. See the "Stub Area" section on page 1-8. 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 the OSPFv2 Feature" section on page 1-13).

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

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **area** *area-id* **stub**
4. (Optional) **area** *area-id* **default-cost** *cost*
5. (Optional) **show ip ospf** *instance-tag*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospf`** *`instance-tag`*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**`area`** *`area-id`* **`stub`**
	Example:
	`switch(config-router)# area 0.0.0.10 stub`
		Creates this area as a stub area.

Step 4	**`area`** *`area-id`* **`default-cost`** *`cost`*
	Example:
	`switch(config-router)# area 0.0.0.10 default-cost 25`
		(Optional) 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	**`show ip ospf`** *`instance-tag`*
	Example:
	`switch(config-if)# show ip ospf 201`
		(Optional) Displays OSPF information.

Step 6	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

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:

Command	Purpose

`area` *`area-id`* **`stub no-summary`**
Example:
`switch(config-router)# area 20 stub no-summary`
	Creates this area as a totally stubby area.

Configuring NSSA

You can configure an NSSA for part of an OSPFv2 domain where limited external traffic is required. See the "Not-So-Stubby Area" section on page 1-9. 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.
- 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.
- No summary—Blocks all summary routes from flooding the NSSA. Use this option on the NSSA ABR.

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

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

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate** [**route-map** *map-name*]] [**no-summary**] [**translate type7** {**always** | **never**} [**suppress-fa**]]
4. (Optional) **area** *area-id* **default-cost** *cost*
5. (Optional) **show ip ospf** *instance-tag*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate**] [**route-map** *map-name*]] [**no-summary**] [**translate type7** {**always** | **never**} [**suppress-fa**]] **Example:** `switch(config-router)# area 0.0.0.10 nssa`	Creates this area as an NSSA.
Step 4	**area** *area-id* **default-cost** *cost* **Example:** `switch(config-router)# area 0.0.0.10 default-cost 25`	(Optional) Sets the cost metric for the default summary route sent into this NSSA.
Step 5	**show ip ospf** *instance-tag* **Example**: `switch(config-if)# show ip ospf 201`	(Optional) Displays OSPF information.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

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 that always translates NSSA External (type 5) LSAs to AS External (type 7) LSAs:

```
switch# configure terminal
switch(config)# router ospf 201
switch(config-router)# area 0.0.0.10 nssa translate type 7 always
switch(config-router)# 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 on page 1-9. 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 the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **area** *area-id* **virtual-link** *router-id*
4. (Optional) **show ip ospf virtual-link** [**brief**]
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**area** *area-id* **virtual-link** *router-id*
	Example:
	`switch(config-router)# area 0.0.0.10 virtual-link 10.1.2.3`
	`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	**show ip ospf virtual-link** [**brief**] **Example**: `switch(config-router-vlink)# show ip ospf virtual-link`	(Optional) Displays OSPF virtual link information.
Step 5	**copy running-config startup-config** **Example:** `switch(config-router-vlink)# copy running-config startup-config`	(Optional) Saves this configuration change.

You can configure the following optional commands in virtual link configuration mode:

Command	Purpose
authentication [**key-chain** *key-id* | **message-digest** | **null**] **Example:** `switch(config-router-vlink)# authentication message-digest`	(Optional) Overrides area-based authentication for this virtual link.
authentication-key [**0** | **3**] *key* **Example:** `switch(config-router-vlink)# authentication-key 0 mypass`	(Optional) 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.
dead-interval *seconds* **Example**: `switch(config-router-vlink)# dead-interval 50`	(Optional) Configures the OSPFv2 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.
hello-interval *seconds* **Example:** `switch(config-router-vlink)# hello-interval 25`	(Optional) Configures the OSPFv2 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.

Command	Purpose
message-digest-key *key-id* **md5** [**0** | **3**] *key* **Example:** `switch(config-router-vlink)# message-digest-key 21 md5 0 mypass`	(Optional) Configures message digest authentication for this virtual link. Use this command if the authentication is set to message-digest. 0 configures the password in cleartext. 3 configures the pass key as 3DES encrypted.
retransmit-interval *seconds* **Example**: `switch(config-router-vlink)# retransmit-interval 50`	(Optional) Configures the OSPFv2 retransmit interval, in seconds. The range is from 1 to 65535. The default is 5.
transmit-delay *seconds* **Example:** `switch(config-router-vlink)# transmit-delay 2`	(Optional) Configures the OSPFv2 transmit-delay, in seconds. The range is from 1 to 450. The default is 1.

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-vlink)# 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-vlink)# copy running-config startup-config
```

Configuring Redistribution

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

You can configure the following optional parameters for route redistribution in OSPF:

- Default information originate—Generates an AS 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.

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

Note If you redistribute static routes, Cisco NX-OS also redistributes the default static route.

Note Redistribution does not work if the access list is used as a **match** option in **route-maps.**

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

Create the necessary route maps used for redistribution.

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **redistribute** {**bgp** *id* | **direct** | **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} **route-map** *map-name*
4. **default-information originate** [**always**] [**route-map** *map-name*]
5. **default-metric** *cost*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospf`** *`instance-tag`*
	
	Example:
	`switch(config)# router ospf 201`
	`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`*
	
	Example:
	`switch(config-router)# redistribute bgp 64496 route-map FilterExternalBGP`
		Redistributes the selected protocol into OSPF through the configured route map.
		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]`
	
	Example:
	`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:
		• **always** —Always generate the default route of 0.0.0. even if the route does not exist in the RIB.
		• **route-map**—Generate the default route if the route map returns true.
		Note This command ignores **match** statements in the route map.

	Command	Purpose

Step 5	**`default-metric`** *`cost`*
	`Example:`
	`switch(config-router)# default-metric 25`
		Sets the cost metric for the redistributed routes. This does not apply to directly connected routes. Use a route map to set the default metric for directly connected routes.

Step 6	**`copy running-config startup-config`**
	`Example:`
	`switch(config-router)# copy running-config startup-config`
		(Optional) Saves this configuration change.

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 will log a warning when that threshold is passed.
- Warning only—Logs a warning only when OSPFv2 reaches the maximum. OSPFv2 continuse to accept redistributed routes.
- Widthdraw—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 the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **redistribute** {**bgp** *id* | **direct**| **eigrp** *id* | **isis** *id* | **ospf** *id* | **rip** *id* | **static**} **route-map** *map-name*
4. **redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]]
5. (Optional) **show running-config ospf**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospf** *instance-tag*
	Example:
	`switch(config)# router ospf 201`
	`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* **Example:** `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*]] **Example:** `switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	Specifies a maximum number of prefixes that OSPFv2 will distribute. The range is from 0 to 65536. Optionally specifies the following: • *threshold*—Percent of maximum prefixes that will trigger a warning message. • **warning-only**—Logs an warning message when the maximum number of prefixes is exceeded. • **withdraw**—Withdraws all redistributed routes. Optionally tries to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* is 60 to 600 seconds. The default is 300 seconds. Use **clear ip ospf redistribution** if all routes are withdrawn.
Step 5	**show running-config ospf** **Example:** `switch(config-router)# show running-config ospf`	(Optional) Displays the OSPFv2 configuration.
Step 6	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Saves this configuration change.

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. See the "Route Summarization" section on page 1-10.

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **area** *area-id* **range** *ip-prefix/length* [**no-advertise**]
4. **summary-address** *ip-prefix/length* [**no-advertise** | **tag** *tag-id*]
5. (Optional) **show ip ospf summary-address**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospf`** *`instance-tag`*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**`area`** *`area-id`* **`range`** *`ip-prefix/length`* [**`no-advertise`**] **Example:** `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 note advertise this summary address in a Network Summary (type 3) LSA.
Step 4	**`summary-address`** *`ip-prefix/length`* [**`no-advertise`** | **`tag`** `tag`] **Example:** `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.

	Command	Purpose
Step 5	**show ip ospf summary-address** **Example:** `switch(config-router)# show ip ospf summary-address`	(Optional) Displays information about OSPF summary addresses.
Step 6	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Saves this configuration change.

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. See the "OSPFv2 Stub Router Advertisements" section on page 1-11.

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.

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **max-metric router-lsa** [**on-startup** [*announce-time*] [**wait-for bgp** *tag*]]
4. (Optional) **copy running-config startup-config**

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

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospf`** *`instance-tag`*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**`max-metric router-lsa`** [**`on-startup`** [*`announce-time`*] [**`wait-for bgp`** *`tag`*]] **Example:** `switch(config-router)# max-metric router-lsa`	Configures OSPFv2 stub route advertisements. On-start-up, advertise when it first comes up or system start time. Wait for BGP to come up.

Step 4	**`copy running-config startup-config`**
	Example:
	`switch(config-router)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to enable the stub router advertisements feature 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
```

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 arriving from a neighbor. LSAs that arrive faster than this time are dropped.
- Pacing LSAs—Set 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" section on page 1-6).
- Throttle LSAs—Set rate limits for generating LSAs. This timer controls how frequently an LSA is generated if no 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" section on page 1-16 for information about the hello interval and dead timer.

BEFORE YOU BEGIN

Ensure that you have enabled the OSPF feature (see the "Enabling the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **timers lsa-arrival** *msec*
4. **timers lsa-group-pacing** *seconds*
5. **timers throttle lsa** *start-time hold-interval max-time*
6. **timers throttle spf** *delay-time hold-time*
7. **interface** *type slot/port*
8. **no switchport**
9. **ip ospf hello-interval** *seconds*
10. **ip ospf dead-interval** *seconds*
11. **ip ospf retransmit-interval** *seconds*
12. **ip ospf transmit-delay** *seconds*
13. (Optional) **show ip ospf**
14. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospf`** *`instance-tag`*
	Example:
	`switch(config)# router ospf 201`
	`switch(config-router)#`
		Creates a new OSPFv2 instance with the configured instance tag.

Step 3	**`timers lsa-arrival`** *`msec`*
	Example:
	`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`*
	Example:
	`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.

	Command	Purpose
Step 5	**timers throttle lsa** *start-time hold-interval max-time* **Example:** `switch(config-router)# timers throttle lsa 3000 6000 6000`	Sets the rate limit in milliseconds for generating LSAs with the following timers: *start-time*—The range is from 50 to 5000 milliseconds. The default value is 50 milliseconds. *hold-interva*l—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds. *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* **Example:** `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*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 8	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 9	**ip ospf hello-interval** *seconds* **Example:** `switch(config-if)# ip ospf retransmit-interval 30`	Sets the hello interval this interface. The range is from 1 to 65535. The default is 10.
Step 10	**ip ospf dead-interval** *seconds* **Example:** `switch(config-if)# ip ospf dead-interval 30`	Sets the dead interval for this interface. The range is from 1 to 65535.
Step 11	**ip ospf retransmit-interval** *seconds* **Example:** `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 12	**ip ospf transmit-delay** *seconds*
	Example:
	`switch(config-if)# ip ospf transmit-delay 450`
	`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 13	**show ip ospf** **Example**: `switch(config-if)# show ip ospf`	(Optional) Displays information about OSPF.
Step 14	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

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
```

Restarting an OSPFv2 Instance

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

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

Command	Purpose
restart ospf *instance-tag* **Example:** `switch(config)# restart ospf 201`	Restarts the OSPFv2 instance and removes all neighbors.

Configuring OSPFv2 with Virtualization

You can configure multiple OSPFv2 instances. You can also create multiple VRFs and use the same or multiple OSPFv2 instances in each VRF. You 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 the OSPFv2 Feature" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf_name*
3. **router ospf** *instance-tag*
4. **vrf** *vrf-name*
5. **maximum-paths** *paths*
6. **interface** *interface-type slot/port*
7. **no switchport**
8. **vrf member** *vrf-name*
9. **ip-address** *ip-prefix/length*
10. **ip router ospf** *instance-tag* **area** *area-id*
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**vrf context** *vrf-name* **Example:** switch(config)# vrf context RemoteOfficeVRF switch(config-vrf)#	Creates a new VRF and enters VRF configuration mode.
Step 3	**router ospf** *instance-tag* **Example:** switch(config-vrf)# router ospf 201 switch(config-router)#	Creates a new OSPFv2 instance with the configured instance tag.
Step 4	**vrf** *vrf-name* **Example:** switch(config-router)# vrf RemoteOfficeVRF switch(config-router-vrf)#	Enters VRF configuration mode.
Step 5	**maximum-paths** *paths* **Example**: switch(config-router-vrf)# maximum-paths 4	(Optional) Configures the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF. Used for load balancing.
Step 6	**interface** *interface-type slot/port* **Example**: switch(config-router-vrf)# interface ethernet 1/2 switch(config-if)#	Enters interface configuration mode. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 7	**no switchport** **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 8	**vrf member** *vrf-name* **Example:** switch(config-if)# vrf member RemoteOfficeVRF	Adds this interface to a VRF.
Step 9	**ip address** *ip-prefix/length* **Example:** 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.

	Command	Purpose

Step 10	**`ip router ospf`** *`instance-tag`* **`area`** *`area-id`*
	Example:
	`switch(config-if)# ip router ospf 201 area 0`
		Assigns this interface to the OSPFv2 instance and area configured.

Step 11	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

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)# no switchport
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)# copy running-config startup-config
```

Verifying the OSPFv2 Configuration

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

Command	Purpose
show ip ospf	Displays the OSPFv2 configuration.
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.
show ip ospf interface *number* [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv2 interface configuration.
show ip ospf lsa-content-changed-list *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.

Command	Purpose
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 VRF-based OSPFv2 configuration.
show running-configuration ospf	Displays the current running OSPFv2 configuration.

Displaying OSPFv2 Statistics

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 ospf policy statistics redistribute {**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

This example shows how to configure OSPFv2:

```
feature ospf
router ospf 201
 router-id 290.0.2.1

interface ethernet 1/2
 no switchport
 ip router ospf 201 area 0.0.0.10
 ip ospf authentication
 ip ospf authentication-key 0 mypass
```

Additional References

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

- Related Documents, page 1-43
- MIBs, page 1-43

Related Documents

Related Topic	Document Title
OSPFv2 CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*
OSPFv3 for IPv6 networks	Chapter 7, "Configuring OSPFv3"
Route maps	Chapter 1, "Configuring Route Policy Manager"

MIBs

MIBs	MIBs Link
• OSPF-MIB • OSPF-TRAP-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

CHAPTER 1

Configuring OSPFv3

This chapter describes how to configure Open Shortest Path First version 3 (OSPFv3) for IPv6 networks on the Cisco NX-OS device.

This chapter includes the following sections:

Information About OSPFv3

OSPFv3 is an IETF link-state protocol (see the "Overview" section on page 1-1). An OSPFv3 router sends a special message, called a *Hello Packet,* out each OSPF-enabled interface to discover other OSPFv3 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 attempt to establish *adjacency*, which means that the routers synchronize their link-state databases to ensure that they have identical OSPFv3 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 OSPFv3 routers eventually have identical link-state databases. When all OSPFv3 routers have identical link-state databases, the network is *converged* (see the "Convergence" section on page 1-6). Each router then uses Dijkstra's Shortest Path First (SPF) algorithm to build its route table.

You can divide OSPFv3 networks into areas. Routers send most LSAs only within one area, which reduces the CPU and memory requirements for an OSPF-enabled router.

OSPFv3 supports IPv6. For information about OSPF for IPv4, see Chapter 1, “Configuring OSPFv3”.

This section includes the following topics:

Comparison of OSPFv3 and OSPFv2

Much of the OSPFv3 protocol is the same as in OSPFv2. OSPFv3 is described in RFC 2740.

The key differences between the OSPFv3 and OSPFv2 protocols are as follows:

- OSPFv3 expands on OSPFv2 to provide support for IPv6 routing prefixes and the larger size of IPv6 addresses.
- LSAs in OSPFv3 are expressed as prefix and prefix length instead of address and mask.
- The router ID and area ID are 32-bit numbers with no relationship to IPv6 addresses.
- OSPFv3 uses link-local IPv6 addresses for neighbor discovery and other features.
- OSPFv3 uses IPv6 for authentication.
- OSPFv3 redefines LSA types.

Hello Packet

OSPFv3 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. OSPFv3 uses Hello packets for the following tasks:

- Neighbor discovery
- Keepalives
- Bidirectional communications
- Designated router election (see the “Designated Routers” section on page 1-4)

The Hello packet contains information about the originating OSPFv3 interface and router, including the assigned OSPFv3 cost of the link, the hello interval, and optional capabilities of the originating router. An OSPFv3 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” section on page 1-3).

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.

OSPFv3 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), the neighbor is removed from the local neighbor table.

Neighbors

An OSPFv3 interface must have a compatible configuration with a remote interface before the two can be considered neighbors. The two OSPFv3 interfaces must match the following criteria:

- Hello interval
- Dead interval
- Area ID (see the "Areas" section on page 1-5)
- Authentication
- Optional capabilities

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

- Neighbor ID—Router ID of the neighbor router.
- Priority—Priority of the neighbor router. The priority is used for designated router election (see the "Designated Routers" section on page 1-4).
- 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 how long since the last Hello packet was received from this neighbor.
- Link-local IPv6 address—Link-local IPv6 address of the neighbor.
- Designated router—Indication of whether the neighbor has been declared as the designated router or backup designated router (see the "Designated Routers" section on page 1-4).
- Local interface—Local interface that received the Hello packet for this neighbor.

When the first Hello packet is received from a new neighbor, the neighbor is entered into the neighbor table in the initialization state. Once bidirectional communication is established, the neighbor state becomes two-way. ExStart and exchange states come next, as the two interfaces exchange their link-state database. Once this is complete, the neighbor moves into the full state, which signifies full adjacency. If the neighbor fails to send any Hello packets in the dead interval, the neighbor is moved to the down state and is no longer considered adjacent.

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" section on page 1-4.

Adjacency is established using Database Description packets, Link State Request packets, and Link State Update packets in OSPFv3. The Database Description packet includes the LSA headers from the link-state database of the neighbor (see the "Link-State Database" section on page 1-7). The local router compares these headers with its own link-state database and determines which LSAs are new or updated.

The local router sends a Link State Request packet for each LSA that it needs new or updated information on. The neighbor responds with a Link State Update packet. This exchange continues until both routers have the same link-state information.

Designated Routers

Networks with multiple routers present a unique situation for OSPFv3. If every router floods the network with LSAs, the same link-state information is sent from multiple sources. Depending on the type of network, OSPFv3 might use a single router, the *designated router* (*DR*), to control the LSA floods and represent the network to the rest of the OSPFv3 area (see the "Areas" section on page 1-5). If the DR fails, OSPFv3 selects a *backup designated router* (BDR). If the DR fails, the BDR becomes the DR.

Network types are as follows:

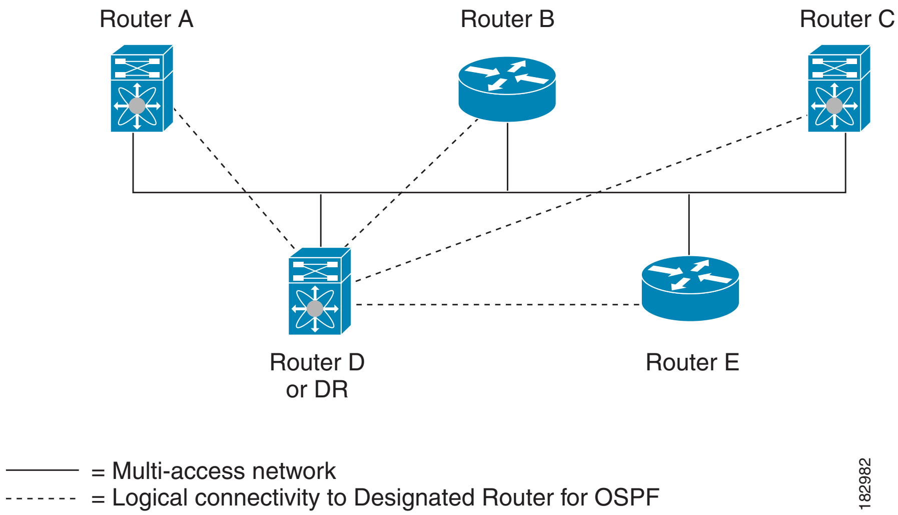
- Point-to-point—A network that exists only between two routers. All neighbors on a point-to-point network establish adjacency and there is no DR.
- Broadcast—A network with multiple routers that can communicate over a shared medium that allows broadcast traffic, such as Ethernet. OSPFv3 routers establish a DR and BDR that controls LSA flooding on the network. OSPFv3 uses the well-known IPv6 multicast addresses, FF02::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 determinant, OSPFv3 chooses the highest router IDs as the DR and BDR.

All other routers establish adjacency with the DR and the BDR and use the IPv6 multicast address FF02::6 to send LSA updates to the DR and BDR. Figure 1-1 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 1-1 DR in Multi-Access Network



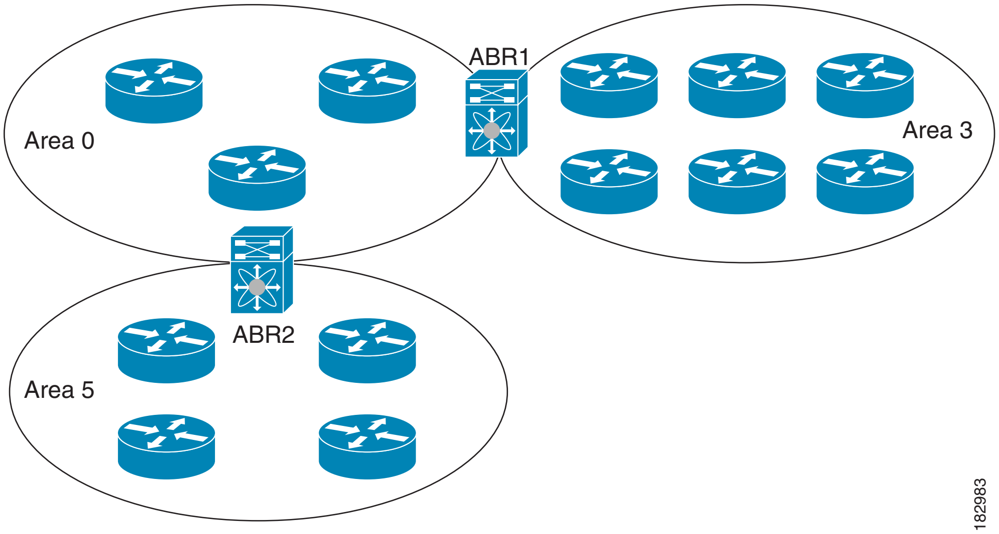
Areas

You can limit the CPU and memory requirements that OSPFv3 puts on the routers by dividing an OSPFv3 network into *areas*. An area is a logical division of routers and links within an OSPFv3 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 can be expressed 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 OSPFv3 network, you must also define the backbone area, which has the reserved area ID of 0. All areas must connect to the backbone area. If you have more than one area, then one or more routers become area border routers (ABRs). An ABR connects to both the backbone area and at least one other defined area (see Figure 1-2).

Figure 1-2 OSPFv3 Areas



The ABR has a separate link-state database for each area which it connects to. The ABR sends Inter-Area Prefix (type 3) LSAs (see the "Route Summarization" section on page 1-10) from one connected area to the backbone area. The backbone area sends summarized information about one area to another area. In Figure 1-2, Area 0 sends summarized information about Area 5 to Area 3.

OSPFv3 defines one other router type: the autonomous system boundary router (ASBR). This router connects an OSPFv3 area to another autonomous system. An autonomous system is a network controlled by a single technical administration entity. OSPFv3 can redistribute its routing information into another autonomous system or receive redistributed routes from another autonomous system. For more information, see the "Advanced Features" section on page 1-8.

Link-State Advertisement

OSPFv3 uses link-state advertisements (LSAs) to build its routing table.

This section includes the following topics:

- LSA Types, page 1-6
- Link Cost, page 1-6
- Flooding and LSA Group Pacing, page 1-7
- Link-State Database, page 1-7

LSA Types

Table 1-1 shows the LSA types supported by Cisco Nexus 6000.

Table 1-1 LSA Types

Type	Name	Description
1	Router LSA	LSA sent by every router. This LSA includes the state and cost of all links but does not include prefix information. Router LSAs trigger an SPF recalculation. Router LSAs are flooded to the local OSPFv3 area.
2	Network LSA	LSA sent by the DR. This LSA lists all routers in the multi-access network but does not include prefix information. Network LSAs trigger an SPF recalculation. See the "Designated Routers" section on page 1-4.
3	Inter-Area Prefix LSA	LSA sent by the area border router to an external area for each destination in local area. This LSA includes the link cost from the border router to the local destination. See the "Areas" section on page 1-5.
4	Inter-Area Router 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 on page 1-5.
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 on page 1-5.
7	Type-7 LSA	LSA generated by the ASBR within an NSSA. This LSA includes the link cost to an external autonomous system destination. Type-7 LSAs are flooded only within the local NSSA. See the "Areas" section on page 1-5.
8	Link LSA	LSA sent by every router, using a link-local flooding scope (see the "Flooding and LSA Group Pacing" section on page 1-7. This LSA includes the link-local address and IPv6 prefixes for this link.
9	Intra-Area Prefix LSA	LSA sent by every router. This LSA includes any prefix or link state changes. Intra-Area Prefix LSAs are flooded to the local OSPFv3 area. This LSA does not trigger an SPF recalculation.
11	Grace LSAs	LSA sent by a restarting router, using a link-local flooding scope. This LSA is used for a graceful restart of OSPFv3. See the "When you configure a summary address, Cisco Nexus 6000 automatically configures a discard route for the summary address to prevent routing black holes and route loops." section on page 1-11.

Link Cost

Each OSPFv3 interface is assigned a *link cost*. The cost is an arbitrary number. By default, Cisco Nexus 6000 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

OSPFv3 floods LSA updates to different sections of the network, depending on the LSA type. OSPFv3 uses the following flooding scopes:

- Link-local—The LSA is flooded only on the local link. Used for Link LSAs and Grace LSAs.
- Area-local—The LSA is flooded throughout a single OSPFv3 area only. Used for Router LSAs, Network LSAs, Inter-Area-Prefix LSAs, Inter-Area-Router LSAs, and Intra-Area-Prefix LSAs.
- AS scope—The LSA is flooded throughout the routing domain. An AS scope is used for AS External LSAs.

LSA flooding guarantees that all routers in the network have identical routing information. LSA flooding depends on the OSPFv3 area configuration (see the "Areas" section on page 1-5). 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 utilization. This feature groups LSAs with similar link-state refresh times to allow OSPFv3 to pack multiple LSAs into an OSPFv3 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 OSPFv3 load on your network.

Link-State Database

Each router maintains a link-state database for the OSPFv3 network. This database contains all the collected LSAs and includes information on all the routes through the network. OSPFv3 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 Nexus 6000 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" section on page 1-7.

Multi-Area Adjacency

OSPFv3 multi-area adjacency allows you to configure a link on the primary interface that is in more than one area. This link becomes the preferred intra-area link in those areas. Multi-area adjacency establishes a point-to-point unnumbered link in an OSPFv3 area that provides a topological path for that area. The primary adjacency uses the link to advertise an unnumbered point-to-point link in the Router LSA for the corresponding area when the neighbor state is full.

The multi-area interface exists as a logical construct over an existing primary interface for OSPFv3; however, the neighbor state on the primary interface is independent of the multi-area interface. The multi-area interface establishes a neighbor relationship with the corresponding multi-area interface on the neighboring router. See the "Configuring Multi-Area Adjacency" section on page 1-25 for more information.

OSPFv3 and the IPv6 Unicast RIB

OSPFv3 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 shortest path for each destination is then put in the OSPFv3 route table. When the OSPFv3 network is converged, this route table feeds into the IPv6 unicast RIB. OSPFv3 communicates with the IPv6 unicast RIB to do the following:

- Add or remove routes
- Handle route redistribution from other protocols
- Provide convergence updates to remove stale OSPFv3 routes and for stub router advertisements (see the "Multiple OSPFv3 Instances" section on page 1-11)

OSPFv3 also runs a modified Dijkstra algorithm for fast recalculation for Inter-Area Prefix, Inter-Area Router, AS-External, type-7, and Intra-Area Prefix (type 3, 4, 5, 7, 8) LSA changes.

Address Family Support

Cisco Nexus 6000 supports multiple address families, such as unicast IPv6 and multicast IPv6. OSPFv3 features that are specific to an *address family* are as follows:

- Default routes
- Route summarization
- Route redistribution
- Filter lists for border routers
- SPF optimization

Use the **address-family ipv6 unicast** command to enter the IPv6 unicast address family configuration mode when configuring these features.

Advanced Features

Cisco Nexus 6000 supports advanced OSPFv3 features that enhance the usability and scalability of OSPFv3 in the network.

This section includes the following topics:

- Stub Area, page 1-9
- Not-So-Stubby Area, page 1-9
- Virtual Links, page 1-10
- Route Redistribution, page 1-10
- Route Summarization, page 1-10
- Multiple OSPFv3 Instances, page 1-11
- SPF Optimization, page 1-11
- Virtualization Support, page 1-11

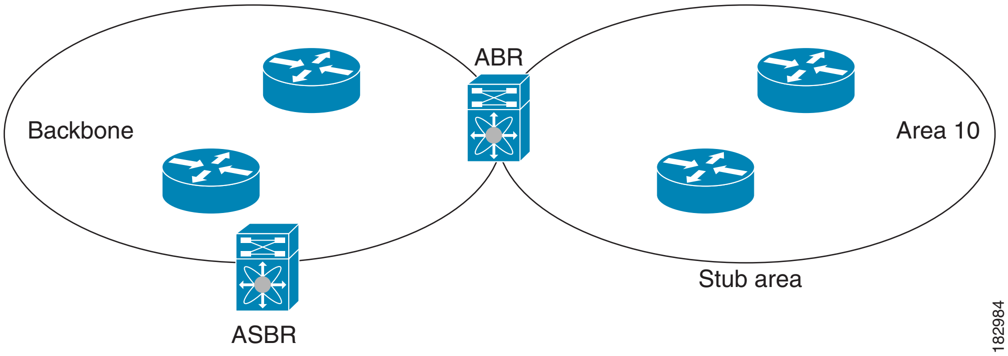
Stub Area

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 on page 1-5). 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. See the "Administrative Distance" section on page 1-7.
- No ASBR routers exist in the stub area.
- You cannot configure virtual links in the stub area.

Figure 1-3 shows an example an OSPFv3 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 1-3 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 an Inter-Area-Prefix LSA with the prefix length set to 0 for IPv6.

Not-So-Stubby Area

A Not-So-Stubby Area (*NSSA*) is similar to the 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 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 type-7 LSA to AS External (type 5) LSAs. The ABR then floods these AS External LSAs throughout the OSPFv3 autonomous system. Summarization and filtering are supported during the translation. See the "Link-State Advertisement" section on page 1-5 for details on type-7 LSAs.

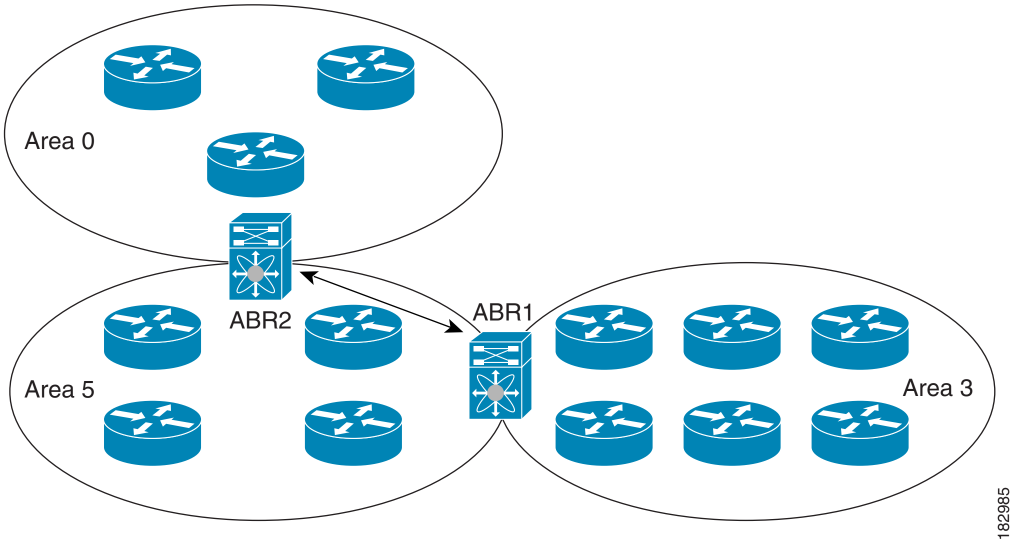
You can, for example, use an NSSA to simplify administration if you are connecting a central site using OSPFv3 to a remote site that is using a different routing protocol. Before an NSSA, the connection between the corporate site border router and a remote router could not be run as an OSPFv3 stub area because routes for the remote site could not be redistributed into a stub area. With an NSSA, you can extend OSPFv3 to cover the remote connection by defining the area between the corporate router and remote router as an NSSA (see the "Configuring NSSA" section on page 1-23).

The backbone Area 0 cannot be an NSSA.

Virtual Links

Virtual links allow you to connect an OSPFv3 area ABR to a backbone area ABR when a direct physical connection is not available. Figure 1-4 shows a virtual link that connects Area 3 to the backbone area through Area 5.

Figure 1-4 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

OSPFv3 can learn routes from other routing protocols by using route redistribution. See the "Route Redistribution" section on page 1-6. You configure OSPFv3 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 OSPFv3. 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 OSPFv3 autonomous system. For more information, see Chapter 1, "Configuring Route Policy Manager,"

Route Summarization

Because OSPFv3 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 2010:11:22:0:1000::1 and 2010:11:22:0:2000:679:1 with one summary address, 2010:11:22::/32.

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, 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 OSPFv3 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 OSPFv3.

When you configure a summary address, Cisco Nexus 6000 automatically configures a discard route for the summary address to prevent routing black holes and route loops.

Multiple OSPFv3 Instances

Cisco Nexus 6000 supports multiple instances of the OSPFv3 protocol. 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 OSPFv3 autonomous system.

The OSPFv3 header includes an instance ID field to identify that OSPFv3 packet for a particular OSPFv3 instance. You can assign the OSPFv3 instance. The interface drops all OSPFv3 packets that do not have a matching OSPFv3 instance ID in the packet header.

Cisco Nexus 6000 allows only one OSPFv3 instance on an interface.

SPF Optimization

Cisco Nexus 6000 optimizes the SPF algorithm in the following ways:

- Partial SPF for Network (type 2) LSAs, Inter-Area Prefix (type 3) LSAs, and AS External (type 5) LSAs—When there is a change on any of these LSAs, Cisco Nexus 6000 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

OSPFv3 supports virtual routing and forwarding (VRF) instances.

Licensing Requirements for OSPFv3

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	OSPFv3 requires a LAN Base Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*.

Prerequisites for OSPFv3

OSPFv3 has the following prerequisites:

- You must be familiar with routing fundamentals to configure OSPFv3.
- You must be logged on to the switch.
- You have configured at least one interface for IPv6 that is capable of communicating with a remote OSPFv3 neighbor.
- You have installed the LAN Base Services license.
- You have completed the OSPFv3 network strategy and planning for your network. For example, you must decide whether multiple areas are required.
- You have enabled OSPFv3 (see the "Enabling OSPFv3" section on page 1-13).
- You are familiar with IPv6 addressing and basic configuration. See Chapter 1, "Configuring IPv6" for information on IPv6 routing and addressing.

Guidelines and Limitations for OSPFv3

OSPFv3 has the following configuration guidelines and limitations:

- You can have up to four instances of OSPFv3 in a VDC.
- Cisco NX-OS displays areas in dotted decimal notation regardless of whether you enter the area in decimal or dotted decimal notation.
- Bidirectional Forwarding Detection (BFD) is not supported for OSPFv3.
- If you configure OSPFv3 in a virtual port channel (vPC) environment, use the following timer commands in router configuration mode on the core switch to ensure fast OSPFv3 convergence when a vPC peer link is shut down:

```
switch (config-router)# timers throttle spf 1 50 50
switch (config-router)# timers lsa-arrival 10
```

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

Table 1-2 lists the default settings for OSPFv3 parameters.

Table 1-2 Default OSPFv3 Parameters

Parameters	Default
Hello interval	10 seconds
Dead interval	40 seconds
Graceful restart grace period	60 seconds
Graceful restart notify period	15 seconds
OSPFv3 feature	Disabled
Stub router advertisement announce time	600 seconds
Reference bandwidth for link cost calculation	40 Gb/s
LSA minimal arrival time	1000 milliseconds
LSA group pacing	10 seconds
SPF calculation initial delay time	0 milliseconds
SPF calculation hold time	5000 milliseconds
SPF calculation initial delay time	0 milliseconds

Configuring Basic OSPFv3

Configure OSPFv3 after you have designed your OSPFv3 network.

This section includes the following topics:

- Enabling OSPFv3, page 1-13
- Creating an OSPFv3 Instance, page 1-14
- Configuring Networks in OSPFv3, page 1-17

Enabling OSPFv3

You must enable OSPFv3 before you can configure OSPFv3.

SUMMARY STEPS

1. **configure terminal**
2. **feature ospfv3**
3. **(Optional) show feature**
4. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**feature ospfv3** **Example:** `switch(config)# feature ospfv3`	Enables OSPFv3.
Step 3	**show feature** **Example:** `switch(config)# show feature`	(Optional) Displays enabled and disabled features.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

To disable the OSPFv3 feature and remove all associated configuration, use the following command in configuration mode.

Command	Purpose
no feature ospfv3 **Example:** `switch(config)# no feature ospfv3`	Disables the OSPFv3 feature and removes all associated configuration.

Creating an OSPFv3 Instance

The first step in configuring OSPFv3 is to create an instance or OSPFv3 instance. You assign a unique instance tag for this OSPFv3 instance. The instance tag can be any string. For each OSPFv3 instance, you can also configure the following optional parameters:

- Router ID—Configures the router ID for this OSPFv3 instance. If you do not use this parameter, the router ID selection algorithm is used. For more information, see the "Router IDs" section on page 1-5.
- Administrative distance—Rates the trustworthiness of a routing information source. For more information, see the "Administrative Distance" section on page 1-7.
- Log adjacency changes—Creates a system message whenever an OSPFv3 neighbor changes its state.
- Maximum paths—Sets the maximum number of equal paths that OSPFv3 installs in the route table for a particular destination. Use this parameter for load balancing between multiple paths.
- Reference bandwidth—Controls the calculated OSPFv3 cost metric for a network. The calculated cost is the reference bandwidth divided by the interface bandwidth. You can override the calculated cost by assigning a link cost when a network is added to the OSPFv3 instance. For more information, see the "Configuring Networks in OSPFv3" section on page 1-17.

For more information about OSPFv3 instance parameters, see the"Configuring Advanced OSPFv3" section on page 1-19.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

Ensure that the OSPFv3 instance tag that you plan on using is not already in use on this router.

Use the **show ospfv3** *instance-tag* command to verify that the instance tag is not in use.

OSPFv3 must be able to obtain a router identifier (for example, a configured loopback address) or you must configure the router ID option.

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **(Optional) router-id** *ip-address*
4. **(Optional) show ipv6 ospfv3** *instance-tag*
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router ospfv3** *instance-tag* **Example:** switch(config)# router ospfv3 201 switch(config-router)#	Creates a new OSPFv3 instance with the configured instance tag.
Step 3	**router-id** *ip-address* **Example:** switch(config-router)# router-id 192.0.2.1	(Optional) Configures the OSPFv3 router ID. This ID uses the dotted decimal notation and identifies this OSPFv3 instance and must exist on a configured interface in the system.
Step 4	**show ipv6 ospfv3** *instance-tag* **Example**: switch(config-router)# show ipv6 ospfv3 201	(Optional) Displays OSPFv3 information.
Step 5	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Saves this configuration change.

To remove the OSPFv3 instance and all associated configuration, use the following command in configuration mode:

Command	Purpose
no router ospfv3 *instance-tag* **Example:** `switch(config)# no router ospfv3 201`	Deletes the OSPFv3 instance and all associated configuration.

Note This command does not remove OSPFv3 configuration in interface mode. You must manually remove any OSPFv3 commands configured in interface mode.

You can configure the following optional parameters for OSPFv3 in router configuration mode:

Command	Purpose
log-adjacency-changes [**detail**] **Example:** `switch(config-router)# log-adjacency-changes`	Generates a system message whenever a neighbor changes state.
passive-interface default **Example:** `switch(config-router)# passive-interface default`	Suppresses routing updates on all interfaces. This command is overridden by the VRF or interface command mode configuration.

You can configure the following optional parameters for OSPFv3 in address family configuration mode:

Command	Purpose
distance *number* **Example:** `switch(config-router-af)# distance 25`	Configures the administrative distance for this OSPFv3 instance. The range is from 1 to 255. The default is 110.
maximum-paths *paths* **Example:** `switch(config-router-af)# maximum-paths 4`	Configures the maximum number of equal OSPFv3 paths to a destination in the route table. The range is from 1 to 64. The default is 8. This command is used for load balancing.

This example shows how to create an OSPFv3 instance with a maximum of four equal OSPFv3 paths per destination:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# maximum-paths 4
switch(config-router)# copy running-config startup-config
```

Configuring Networks in OSPFv3

You can configure a network to OSPFv3 by associating it through the interface that the router uses to connect to that network (see the "Neighbors" section on page 1-3). 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 OSPFv3 is not enabled on an interface until you configure a valid IPv6 address for that interface.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **ipv6 address** *ipv6-prefix/length*
4. **ipv6 router ospfv3** *instance-tag* **area** *area-id* [**secondaries none**]
5. **(Optional) show ipv6 ospfv3** *instance-tag* **interface** *interface-type slot/port*
6. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`interface-type slot/port`*
	
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**`ipv6 address`** *`ipv6-prefix/length`*
	
	Example:
	`switch(config-if)# ipv6 address`
	`2001:0DB8::1/48`
		Assigns an IPv6 address to this interface.

	Command	Purpose
Step 4	**ipv6 router ospfv3** *instance-tag* **area** *area-id* [**secondaries none**] **Example:** `switch(config-if)# ipv6 router ospfv3 201 area 0`	Adds the interface to the OSPFv3 instance and area.
Step 5	**show ipv6 ospfv3** *instance-tag* **interface** *interface-type slot/port* **Example**: `switch(config-if)# show ipv6 ospfv3 201 interface ethernet 1/2`	(Optional) Displays OSPFv3 information. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 6	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

You can configure the following optional parameters for OSPFv3 in interface configuration mode:

Command	Purpose
ospfv3 cost *number* **Example:** `switch(config-if)# ospfv3 cost 25`	Configures the OSPFv3 cost metric for this interface. The default is to calculate a cost metric, based on the reference bandwidth and interface bandwidth. The range is from 1 to 65535.
ospfv3 dead-interval *seconds* **Example:** `switch(config-if)# ospfv3 dead-interval 50`	Configures the OSPFv3 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.
ospfv3 hello-interval *seconds* **Example:** `switch(config-if)# ospfv3 hello-interval 25`	Configures the OSPFv3 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.
ospfv3 instance *instance* **Example:** `switch(config-if)# ospfv3 instance 25`	Configures the OSPFv3 instance ID. The range is from 0 to 255. The default is 0. The instance ID is link-local in scope.
ospfv3 mtu-ignore **Example:** `switch(config-if)# ospfv3 mtu-ignore`	Configures OSPFv3 to ignore any IP maximum transmission unit (MTU) mismatch with a neighbor. The default is to not establish adjacency if the neighbor MTU does not match the local interface MTU.

ospfv3 network {**broadcast** | **point-point**}
Example:
`switch(config-if)# ospfv3 network broadcast`
	Sets the OSPFv3 network type.

Command	Purpose
`[default	no] ospfv3 passive-interface` **Example:** `switch(config-if)# ospfv3 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.
ospfv3 priority *number* **Example:** `switch(config-if)# ospfv3 priority 25`	Configures the OSPFv3 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 on page 1-4.
ospfv3 shutdown **Example:** `switch(config-if)# ospfv3 shutdown`	Shuts down the OSPFv3 instance on this interface.

This example shows how to add a network area 0.0.0.10 in OSPFv3 instance 201:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ipv6 address 2001:0DB8::1/48
switch(config-if)# ipv6 ospfv3 201 area 0.0.0.10
switch(config-if)# copy running-config startup-config
```

Configuring Advanced OSPFv3

Configure OSPFv3 after you have designed your OSPFv3 network.

This section includes the following topics:

- Configuring Filter Lists for Border Routers, page 1-20
- Configuring Stub Areas, page 1-21
- Configuring a Totally Stubby Area, page 1-22
- Configuring NSSA, page 1-23
- Configuring Multi-Area Adjacency, page 1-25
- Configuring Virtual Links, page 1-26
- Configuring Redistribution, page 1-28
- Limiting the Number of Redistributed Routes, page 1-30
- Configuring Route Summarization, page 1-32
- Modifying the Default Timers, page 1-34
- Configuring Graceful Restart, page 1-36
- Restarting an OSPFv3 Instance, page 1-37
- Configuring OSPFv3 with Virtualization, page 1-38

Configuring Filter Lists for Border Routers

You can separate your OSPFv3 domain into a series of areas that contain related networks. All areas must connect to the backbone area through an area border router (ABR). OSPFv3 domains can connect to external domains as well through an autonomous system border router (ASBR). See the "Areas" section on page 1-5.

ABRs have the following optional configuration parameters:

- Area range—Configures route summarization between areas. For more information, see the "Configuring Route Summarization" section on page 1-32.
- Filter list—Filters the Inter-Area Prefix (type 3) LSAs that are allowed in from an external area on an ABR.

ASBRs also support filter lists.

BEFORE YOU BEGIN

Create the route map that the filter list uses to filter IP prefixes in incoming or outgoing Inter-Area Prefix (type 3) LSAs. See Chapter 1, "Configuring Route Policy Manager."

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **address-family ipv6 unicast**
4. **area** *area-id* **filter-list route-map** *map-name* {**in** | **out**}
5. **(Optional) show ipv6 ospfv3 policy statistics area** *id* **filter-list** {**in** | **out**}
6. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	`router ospfv3 instance-tag`
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	`address-family ipv6 unicast`
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

	Command	Purpose

Step 4	**area** *area-id* **filter-list route-map** *map-name* {**in** | **out**}
	Example:
	`switch(config-router-af)# area 0.0.0.10 filter-list route-map FilterLSAs in`
		Filters incoming or outgoing Inter-Area Prefix (type 3) LSAs on an ABR.

Step 5	**show ipv6 ospfv3 policy statistics area** *id* **filter-list** {**in** | **out**}
	Example:
	`switch(config-if)# show ipv6 ospfv3 policy statistics area 0.0.0.10 filter-list in`
		(Optional) Displays OSPFv3 policy information.

Step 6	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure a filter list for a border router:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# area 0.0.0.10 filter-list route-map FilterLSAs in
switch(config-router-af)# copy running-config startup-config
```

Configuring Stub Areas

You can configure a stub area for part of an OSPFv3 domain where external traffic is not necessary. Stub areas block AS External (type 5) LSAs, limiting unnecessary routing to and from selected networks. See the "Stub Area" section on page 1-9. You can optionally block all summary routes from going into the stub area.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

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

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **area** *area-id* **stub**
4. **(Optional) address-family ipv6 unicast**
5. **(Optional) area** *area-id* **default-cost** *cost*
6. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospfv3`** *`instance-tag`*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**`area`** *`area-id`* **`stub`**
	Example:
	`switch(config-router)# area 0.0.0.10 stub`
		Creates this area as a stub area.

Step 4	**`address-family ipv6 unicast`**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		(Optional) Enters IPv6 unicast address family mode.

Step 5	**`area`** *`area-id`* **`default-cost`** *`cost`*
	Example:
	`switch(config-router-af)# area 0.0.0.10 default-cost 25`
		(Optional) Sets the cost metric for the default summary route sent into this stub area. The range is from 0 to 16777215.

Step 6	**`copy running-config startup-config`**
	Example:
	`switch(config-router)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This shows how to create a stub area that blocks all summary route updates:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 stub no-summary
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:

Command	Purpose

`area` *`area-id`* **`stub no-summary`**
Example:
`switch(config-router)# area 20 stub no-summary`
	Creates this area as a totally stubby area.

Configuring NSSA

You can configure an NSSA for part of an OSPFv3 domain where limited external traffic is required. See the "Not-So-Stubby Area" section on page 1-9. You can optionally translate this external traffic to an AS External (type 5) LSA and flood the OSPFv3 domain with this routing information. An NSSA can be configured with the following optional parameters:

- No redistribution—Redistributes routes that bypass the NSSA to other areas in the OSPFv3 autonomous system. Use this option when the NSSA ASBR is also an ABR.
- Default information originate—Generates a 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 you want are flooded throughout the NSSA and other areas.
- Translate—Translates Type-7 LSAs to AS External (type 5) LSAs for areas outside the NSSA. Use this command on an NSSA ABR to flood the redistributed routes throughout the OSPFv3 autonomous system. You can optionally suppress the forwarding address in these AS External LSAs.
- No summary—Blocks all summary routes from flooding the NSSA. Use this option on the NSSA ABR.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

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

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate]** [**route-map** *map-name*] [**no-summary**] [**translate type7** {**always** | **never**} [**suppress-fa**]]
4. **(Optional) address-family ipv6 unicast**
5. **(Optional) area** *area-id* **default-cost** *cost*
6. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**area** *area-id* **nssa** [**no-redistribution**] [**default-information-originate**] [**route-map** *map-name*] [**no-summary**] [**translate type7** {**always** | **never**} [**suppress-fa**]] **Example:** `switch(config-router)# area 0.0.0.10 nssa`	Creates this area as an NSSA.

Step 4	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		(Optional) Enters IPv6 unicast address family mode.

Step 5	**area** *area-id* **default-cost** *cost* **Example:** `switch(config-router-af)# area 0.0.0.10 default-cost 25`	(Optional) Sets the cost metric for the default summary route sent into this NSSA. The range is from 0 to 16777215.
Step 6	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# router ospfv3 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 ospfv3 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 ospfv3 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 that always translates Type-7 LSAs to AS External (type 5) LSAs:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa translate type 7 always
switch(config-router)# copy running-config startup-config
```

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

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 nssa no-summary
switch(config-router)# copy running-config startup-config
```

Configuring Multi-Area Adjacency

You can add more than one area to an existing OSPFv3 interface. The additional logical interfaces support multi-area adjacency.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

Ensure that you have configured a primary area for the interface (see the "Configuring Networks in OSPFv3" section on page 1-17.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **ipv6 router ospfv3** *instance-tag* **multi-area** *area-id*
4. **(Optional) show ipv6 ospfv3** *instance-tag* **interface** *interface-type slot/port*
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`interface-type slot/port`*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**`ipv6 router ospfv3`** *`instance-tag`* **`multi-area`** *`area-id`*
	Example:
	`switch(config-if)# ipv6 router ospfv3 201 multi-area 3`
		Adds the interface to another area.

Step 4	**`show ipv6 ospfv3`** *`instance-tag`* **`interface`** *`interface-type slot/port`*
	Example:
	`switch(config-if)# show ipv6 ospfv3 201 interface ethernet 1/2`
		(Optional) Displays OSPFv3 information.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 5	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to add a second area to an OSPFv3 interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ipv6 address 2001:0DB8::1/48
switch(config-if)# ipv6 ospfv3 201 area 0.0.0.10
switch(config-if)# ipv6 ospfv3 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 on page 1-10. You can configure the following optional parameters for a virtual link:

- Authentication—Sets 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.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **area** *area-id* **virtual-link** *router-id*
4. **(Optional) show ipv6 ospfv3 virtual-link** [**brief**]
5. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospfv3`** *`instance-tag`*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**`area`** *`area-id`* **`virtual-link`** *`router-id`*
	Example:
	`switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::1`
	`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	**`show ipv6 ospfv3 virtual-link`** `[`**`brief`**`]`
	Example:
	`switch(config-if)# show ipv6 ospfv3 virtual-link`
		(Optional) Displays OSPFv3 virtual link information.

Step 5	**`copy running-config startup-config`**
	Example:
	`switch(config-router)# copy running-config startup-config`
		(Optional) Saves this configuration change.

You can configure the following optional commands in virtual link configuration mode:

Command	Purpose
dead-interval *seconds* **Example**: `switch(config-router-vlink)# dead-interval 50`	(Optional) Configures the OSPFv3 dead interval, in seconds. The range is from 1 to 65535. The default is four times the hello interval, in seconds.
hello-interval *seconds* **Example:** `switch(config-router-vlink)# hello-interval 25`	(Optional) Configures the OSPFv3 hello interval, in seconds. The range is from 1 to 65535. The default is 10 seconds.
retransmit-interval *seconds* **Example**: `switch(config-router-vlink)# retransmit-interval 50`	(Optional) Configures the OSPFv3 retransmit interval, in seconds. The range is from 1 to 65535. The default is 5.
transmit-delay *seconds* **Example:** `switch(config-router-vlink)# transmit-delay 2`	(Optional) Configures the OSPFv3 transmit-delay, in seconds. The range is from 1 to 450. The default is 1.

These examples show how to create a simple virtual link between two ABRs:

Configuration for ABR 1 (router ID 2001:0DB8::1) is as follows:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::10
switch(config-router)# copy running-config startup-config
```

Configuration for ABR 2 (router ID 2001:0DB8::10) is as follows:

```
switch# configure terminal
switch(config)# router ospfv3 101
switch(config-router)# area 0.0.0.10 virtual-link 2001:0DB8::1
switch(config-router)# copy running-config startup-config
```

Configuring Redistribution

You can redistribute routes learned from other routing protocols into an OSPFv3 autonomous system through the ASBR.

You can configure the following optional parameters for route redistribution in OSPFv3:

- Default information originate—Generates an AS 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.

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

Note If you redistribute static routes, Cisco NX-OS also redistributes the default static route.

Note Redistribution does not work if the access list is used as a **match** option in **route-maps.**

BEFORE YOU BEGIN

Create the necessary route maps used for redistribution.

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **address-family ipv6 unicast**
4. **redistribute** {**bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static**} **route-map** *map-name*
5. **default-information originate** [**always**] [**route-map** *map-name*]
6. **default-metric** *cost*
7. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router ospfv3`** *`instance-tag`*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**`address-family ipv6 unicast`**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 4	**`redistribute`** {**`bgp`** *`id`* | **`direct`** | **`isis`** *`id`* | **`rip`** *`id`* | **`static`**} **`route-map`** *`map-name`*
	Example:
	`switch(config-router-af)# redistribute bgp route-map FilterExternalBGP`
		Redistributes the selected protocol into OSPFv3 through the configured route map.
		Note If you redistribute static routes, Cisco NX-OS also redistributes the default static route.

	Command	Purpose
Step 5	**default-information originate** [**always**] [**route-map** map-name] **Example:** switch(config-router-af)# default-information-originate route-map DefaultRouteFilter	Creates a default route into this OSPFv3 domain if the default route exists in the RIB. Use the following optional keywords: • **always** —Always generates the default route of 0.0.0. even if the route does not exist in the RIB. • **route-map**—Generates the default route if the route map returns true. **Note** This command ignores **match** statements in the route map.
Step 6	**default-metric** *cost* **Example:** switch(config-router-af)# default-metric 25	Sets the cost metric for the redistributed routes. The range is from 1 to 16777214. This command does not apply to directly connected routes. Use a route map to set the default metric for directly connected routes.
Step 7	**copy running-config startup-config** **Example:** switch(config-router)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to redistribute the Border Gateway Protocol (BGP) into OSPFv3:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# redistribute bgp route-map FilterExternalBGP
switch(config-router-af)# copy running-config startup-config
```

Limiting the Number of Redistributed Routes

Route redistribution can add many routes to the OSPFv3 route table. You can configure a maximum limit to the number of routes accepted from external protocols. OSPFv3 provides the following options to configure redistributed route limits:

- Fixed limit—Logs a message when OSPFv3 reaches the configured maximum. OSPFv3 does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where OSPFv3 logs a warning when that threshold is passed.
- Warning only—Logs a warning only when OSPFv3 reaches the maximum. OSPFv3 continues to accept redistributed routes.
- Withdraw—Starts the configured timeout period when OSPFv3 reaches the maximum. After the timeout period, OSPFv3 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, OSPFv3 withdraws all redistributed routes. You must clear this condition before OSPFv3 accepts more redistributed routes. You can optionally configure the timeout period.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **address-family ipv6 unicast**
4. **redistribute** {**bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static**} **route-map** *map-name*
5. **redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]]
6. **(Optional) show running-config ospfv3**
7. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 4	**redistribute** {**bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static**} **route-map** *map-name* **Example:** `switch(config-router-af)# redistribute bgp route-map FilterExternalBGP`	Redistributes the selected protocol into OSPFv3 through the configured route map.
Step 5	**redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timemout*]] **Example:** `switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`	Specifies a maximum number of prefixes that OSPFv3 distributes. The range is from 0 to 65536. Optionally, specifies the following: • *threshold*—Percent of maximum prefixes that triggers a warning message. • **warning-only**—Logs an warning message when the maximum number of prefixes is exceeded. • **withdraw**—Withdraws all redistributed routes and optionally tries to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* range is from 60 to 600 seconds. The default is 300 seconds.

	Command	Purpose
Step 6	**show running-config ospfv3** **Example:** `switch(config-router)# show running-config ospfv3`	(Optional) Displays the OSPFv3 configuration.
Step 7	**copy running-config startup-config** **Example:** `switch(config-router)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to limit the number of redistributed routes into OSPFv3:

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)# redistribute bgp route-map FilterExternalBGP
switch(config-router-af)# 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" section on page 1-10.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **address-family ipv6 unicast**
4. **area** *area-id* **range** *ipv6-prefix/length* [**no-advertise**] [**cost** *cost*]

   or
5. **summary-address** *ipv6-prefix/length* [**no-advertise**] [**tag** *tag*]
6. **(Optional) show ipv6 ospfv3 summary-address**
7. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router ospfv3** *instance-tag* **Example:** switch(config)# router ospfv3 201 switch(config-router)#	Creates a new OSPFv3 instance with the configured instance tag.
Step 3	**address-family ipv6 unicast** **Example:** switch(config-router)# address-family ipv6 unicast switch(config-router-af)#	Enters IPv6 unicast address family mode.
Step 4	**area** *area-id* **range** *ipv6-prefix/length* [**no-advertise**] [**cost** *cost*] **Example:** switch(config-router-af)# area 0.0.0.10 range 2001:0DB8::/48 advertise	Creates a summary address on an ABR for a range of addresses *and o*ptionally advertises this summary address in a Inter-Area Prefix (type 3) LSA. The *cost* range is from 0 to 16777215.
Step 5	**summary-address** *ipv6-prefix/length* [**no-advertise**][**tag** *tag*] **Example:** switch(config-router-af)# summary-address 2001:0DB8::/48 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 6	**show ipv6 ospfv3 summary-address** **Example**: switch(config-router)# show ipv6 ospfv3 summary-address	(Optional) Displays information about OSPFv3 summary addresses.
Step 7	**copy running-config startup-config** **Example:** switch(config-router)# copy running-config startup-config	(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router)# area 0.0.0.10 range 2001:0DB8::/48
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 ospfv3 201
switch(config-router)# address-family ipv6 unicast
switch(config-router)# summary-address 2001:0DB8::/48
switch(config-router)# copy running-config startup-config
```

Modifying the Default Timers

OSPFv3 includes a number of timers that control the behavior of protocol messages and shortest path first (SPF) calculations. OSPFv3 includes the following optional timer parameters:

- LSA arrival time—Sets the minimum interval allowed between LSAs arriving 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" section on page 1-7).
- Throttle LSAs—Sets 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 OSPFv3" section on page 1-17 for information on the hello interval and dead timer.

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **timers lsa-arrival** *msec*
4. **timers lsa-group-pacing** *seconds*
5. **timers throttle lsa** *start-time hold-interval max-time*
6. **address-family ipv6 unicast**
7. **timers throttle spf** *delay-time hold-time*
8. **interface** *type slot/port*
9. **ospfv3 retransmit-interval** *seconds*
10. **ospfv3 transmit-delay** *seconds*
11. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router ospfv3** *instance-tag*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 3	**timers lsa-arrival** *msec* **Example:** `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* **Example:** `switch(config-router)# timers lsa-group-pacing 200`	Sets the interval in seconds for grouping LSAs. The range is from 1 to 1800. The default is 10 seconds.
Step 5	**timers throttle lsa** *start-time hold-interval max-time* **Example:** `switch(config-router)# timers throttle lsa network 350 5000 6000`	Sets the rate limit in milliseconds for generating LSAs. You can configure the following timers: *start-time*—The range is from 50 to 5000 milliseconds. The default value is 50 milliseconds. *hold-interva*l—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds. *max-time*—The range is from 50 to 30,000 milliseconds. The default value is 5000 milliseconds.

Step 6	**address-family ipv6 unicast**
	Example:
	`switch(config-router)# address-family ipv6 unicast`
	`switch(config-router-af)#`
		Enters IPv6 unicast address family mode.

Step 7	**timers throttle spf** *delay-time hold-time* **Example:** `switch(config-router)# timers throttle spf 3000 2000`	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 8	**interface** *type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 9	**ospfv3 retransmit-interval** *seconds* **Example:** `switch(config-if)# ospfv3 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.

	Command	Purpose

Step 10	**ospfv3 transmit-delay** *seconds*
	Example:
	`switch(config-if)# ospfv3 transmit-delay 600`
	`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 11	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# router ospfv3 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 OSPFv3 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 OSPFv3 instance. OSPFv3 does not participate in the graceful restart of a neighbor.
- Planned graceful restart only—Configures OSPFv3 to support graceful restart only in the event of a planned restart.

BEFORE YOU BEGIN

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

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

SUMMARY STEPS

1. **configure terminal**
2. **router ospfv3** *instance-tag*
3. **graceful-restart**
4. **graceful-restart grace-period** *seconds*
5. **graceful-restart helper-disable**
6. **graceful-restart planned-only**
7. **(Optional) show ipv6 ospfv3** *instance-tag*
8. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router ospfv3** *instance-tag* **Example:** switch(config)# router ospfv3 201 switch(config-router)#	Creates a new OSPFv3 instance with the configured instance tag.
Step 3	**graceful-restart** **Example:** switch(config-router)# graceful-restart	Enables graceful restart. A graceful restart is enabled by default.
Step 4	**graceful-restart grace-period** *seconds* **Example:** 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	**graceful-restart helper-disable** **Example:** switch(config-router)# graceful-restart helper-disable	Disables helper mode. Enabled by default.
Step 6	**graceful-restart planned-only** **Example:** switch(config-router)# graceful-restart planned-only	Configures graceful restart for planned restarts only.
Step 7	**show ipv6 ospfv3** *instance-tag* **Example**: switch(config-if)# show ipv6 ospfv3 201	(Optional) Displays OSPFv3 information.
Step 8	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# router ospfv3 201
switch(config-router)# graceful-restart
switch(config-router)# graceful-restart grace-period 120
switch(config-router)# copy running-config startup-config
```

Restarting an OSPFv3 Instance

You can restart an OSPv3 instance. This action clears all neighbors for the instance.

To restart an OSPFv3 instance and remove all associated neighbors, use the following command:

Command	Purpose

`restart ospfv3 instance-tag`
Example:
`switch(config)# restart ospfv3 201`
	Restarts the OSPFv3 instance and removes all neighbors.

Configuring OSPFv3 with Virtualization

You can configure multiple OSPFv3 instances. You can also create multiple VRFs and use the same or multiple OSPFv3 instances in each VRF. You assign an OSPFv3 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

You must enable OSPFv3 and create the OSPFv3 instance (see the "Enabling OSPFv3" section on page 1-13).

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf_name*
3. **router ospfv3** *instance-tag*
4. **vrf** *vrf-name*
5. (Optional) **maximum-paths** *paths*
6. **interface** *type slot/port*
7. **vrf member** *vrf-name*
8. **ipv6 address** *ipv6-prefix/length*
9. **ipv6 ospfv3** *instance-tag* **area** *area-id*
10. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`vrf context`** *`vrf-name`*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

Step 3	**`router ospfv3`** *`instance-tag`*
	Example:
	`switch(config)# router ospfv3 201`
	`switch(config-router)#`
		Creates a new OSPFv3 instance with the configured instance tag.

Step 4	**`vrf`** *`vrf-name`*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters VRF configuration mode.

Step 5	**`maximum-paths`** *`paths`*
	Example:
	`switch(config-router-vrf)# maximum-paths 4`
		(Optional) Configures the maximum number of equal OSPFv3 paths to a destination in the route table for this VRF. Use this command for load balancing.

Step 6	**`interface`** *`type slot/port`*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot*/*port* syntax is *slot*/*QSFP-module*/*port*.

Step 7	**`vrf member`** *`vrf-name`*
	Example:
	`switch(config-if)# vrf member RemoteOfficeVRF`
		Adds this interface to a VRF.

Step 8	**`ipv6 address`** *`ipv6-prefix/length`*
	Example:
	`switch(config-if)# ipv6 address 2001:0DB8::1/48`
		Configures an IP address for this interface. You must do this step after you assign this interface to a VRF.

Step 9	**`ipv6 ospfv3`** *`instance-tag`* **`area`** *`area-id`*
	Example:
	`switch(config-if)# ipv6 ospfv3 201 area 0`
		Assigns this interface to the OSPFv3 instance and area configured.

Step 10	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

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-vrf)# exit
switch(config)# router ospfv3 201
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# vrf member NewVRF
switch(config-if)# ipv6 address 2001:0DB8::1/48
switch(config-if)# ipv6 ospfv3 201 area 0
switch(config-if)# copy running-config startup-config
```

Verifying the OSPFv3 Configuration

To display the OSPFv3 configuration, perform one of the following tasks:

Command	Purpose
show ipv6 ospfv3	Displays the OSPFv3 configuration.
show ipv6 ospfv3 border-routers	Displays the internal OSPFv3 routing table entries to an ABR and ASBR.
show ipv6 ospfv3 database	Displays lists of information related to the OSPFv3 database for a specific router.
show ipv6 ospfv3 interface *type number* [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 interface configuration.
show ipv6 ospfv3 neighbors	Displays the neighbor information. Use the **clear ospfv3 neighbors** command to remove adjacency with all neighbors.
show ipv6 ospfv3 request-list	Displays a list of LSAs requested by a router.
show ipv6 ospfv3 retransmission-list	Displays a list of LSAs waiting to be retransmitted.
show ipv6 ospfv3 summary-address	Displays a list of all summary address redistribution information configured under an OSPFv3 instance.
show running-configuration ospfv3	Displays the current running OSPFv3 configuration.

Monitoring OSPFv3

To display OSPFv3 statistics, use the following commands:

Command	Purpose
show ipv6 ospfv3 memory	Displays the OSPFv3 memory usage statistics.
show ipv6 ospfv3 policy statistics area *area-id* **filter-list** {**in** | **out**} [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 route policy statistics for an area.
show ipv6 ospfv3 policy statistics redistribute {**bgp** *id* | **direct** | **isis** *id* | **rip** *id* | **static**} **vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 route policy statistics.
show ipv6 ospfv3 statistics [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 event counters.
show ipv6 ospfv3 traffic [*interface*-*type number*] [**vrf** {*vrf-name* | **all** | **default** | **management**}]	Displays the OSPFv3 packet counters.

Configuration Examples for OSPFv3

This example shows how to configure OSPFv3:

```
feature ospfv3
router ospfv3 201
 router-id 290.0.2.1

interface ethernet 1/2
 ipv6 address 2001:0DB8::1/48
 ipv6 ospfv3 201 area 0.0.0.10
```

Related Topics

The following topics can give more information on OSPFv3:

- Chapter 1, “Configuring OSPFv3”
- Chapter 1, “Configuring Route Policy Manager”

Additional References

For additional information related to implementing OSPFv3, see the following sections:

- Related Documents, page 1-42
- Related Documents, page 1-42

Related Documents

Related Topic	Document Title
OSPFv3 CLI commands	*Cisco Nexus 6000 Series NX-OS Unicast Routing Command Reference, Release 6.x*

Configuring EIGRP

This chapter describes how to configure the Enhanced Interior Gateway Routing Protocol (*EIGRP*) on the Cisco NX-OS switch.

This chapter includes the following sections:

Information About EIGRP

EIGRP combines the benefits of distance vector protocols with the features of link-state protocols. EIGRP sends out periodic hello messages for neighbor discovery. Once EIGRP learns a new neighbor, it sends a one-time update of all the local EIGRP routes and route metrics. The receiving EIGRP router calculates the route distance based on the received metrics and the locally assigned cost of the link to that neighbor. After this initial full route table update, EIGRP sends incremental updates to only those neighbors affected by the route change. This process speeds convergence and minimizes the bandwidth used by EIGRP.

This section includes the following topics:

EIGRP Components

EIGRP has the following basic components:

- Reliable Transport Protocol, page 1-2
- Neighbor Discovery and Recovery, page 1-2
- Diffusing Update Algorithm, page 1-2

Reliable Transport Protocol

The *Reliable Transport Protocol* guarantees ordered delivery of EIGRP packets to all neighbors. (See the "Neighbor Discovery and Recovery" section on page 1-2.) The Reliable Transport Protocol supports an intermixed transmission of multicast and unicast packets. The reliable transport can send multicast packets quickly when unacknowledged packets are pending. This provision helps to ensure that the convergence time remains low for various speed links. See the "Configuring Advanced EIGRP" section on page 1-13 for details about modifying the default timers that control the multicast and unicast packet transmissions.

The Reliable Transport Protocol includes the following message types:

- Hello—Used for neighbor discovery and recovery. By default, EIGRP sends a periodic multicast hello message on the local network at the configured *hello interval*. By default, the hello interval is 5 seconds.
- Acknowledgement—Verifies reliable reception of Updates, Queries, and Replies.
- Updates—Sends to affected neighbors when routing information changes. Updates include the route destination, address mask, and route metrics such as delay and bandwidth. The update information is stored in the EIGRP topology table.
- Queries and Replies—Sent as necessary as part of the Diffusing Update Algorithm used by EIGRP.

Neighbor Discovery and Recovery

EIGRP uses the hello messages from the Reliable Transport Protocol to discover neighboring EIGRP routers on directly attached networks. EIGRP adds neighbors to the neighbor table. The information in the neighbor table includes the neighbor address, the interface it was learned on, and the *hold time*, which indicates how long EIGRP should wait before declaring a neighbor unreachable. By default, the hold time is three times the hello interval or 15 seconds.

EIGRP sends a series of Update messages to new neighbors to share the local EIGRP routing information. This route information is stored in the EIGRP topology table. After this initial transmission of the full EIGRP route information, EIGRP sends Update messages only when a routing change occurs. These Update messages contain only the new or changed information and are sent only to the neighbors affected by the change. See the "EIGRP Route Updates" section on page 1-3.

EIGRP also uses the Hello messages as a keepalive to its neighbors. As long as hello messages are received, Cisco NX-OS can determine that a neighbor is alive and functioning.

Diffusing Update Algorithm

The *Diffusing Update Algorithm* (DUAL) calculates the routing information based on the destination networks in the topology table. The topology table includes the following information:

- IPv4 address/mask—The network address and network mask for this destination.

- Successors—The IP address and local interface connection for all *feasible successors* or neighbors that advertise a shorter distance to the destination than the current *feasible distance*.
- Feasibility distance (FD)—The lowest calculated distance to the destination. The feasibility distance is the sum of the advertised distance from a neighbor plus the cost of the link to that neighbor.

DUAL uses the distance metric to select efficient, loop-free paths. DUAL selects routes to insert into the unicast Routing Information Base (RIB) based on feasible successors. When a topology change occurs, DUAL looks for feasible successors in the topology table. If there are feasible successors, DUAL selects the feasible successor with the lowest feasible distance and inserts that into the unicast RIB, avoiding unnecessary recomputation.

When there are no feasible successors but there are neighbors advertising the destination, DUAL transitions from the passive state to the active state and triggers a recomputation to determine a new successor or next-hop router to the destination. The amount of time required to recompute the route affects the convergence time. EIGRP sends Query messages to all neighbors, searching for feasible successors. Neighbors that have a feasible successor send a Reply message with that information. Neighbors that do not have feasible successors trigger a DUAL recomputation.

EIGRP Route Updates

When a topology change occurs, EIGRP sends an Update message with only the changed routing information to affected neighbors. This Update message includes the distance information to the new or updated network destination.

The distance information in EIGRP is represented as a composite of available route metrics, including bandwidth, delay, load utilization, and link reliability. Each metric has an associated weight that determines if the metric is included in the distance calculation. You can configure these metric weights. You can fine-tune link characteristics to achieve optimal paths, but we recommend that you use the default settings for most configurable metrics.

This section includes the following topics:

- Internal Route Metrics, page 1-3
- External Route Metrics, page 1-4
- EIGRP and the Unicast RIB, page 1-4

Internal Route Metrics

Internal routes are routes that occur between neighbors within the same EIGRP autonomous system. These routes have the following metrics:

- Next hop—The IP address of the next-hop router.
- Delay—The sum of the delays configured on the interfaces that make up the route to the destination network. Configured in tens of microseconds.
- Bandwidth—The calculation from the lowest configured bandwidth on an interface that is part of the route to the destination.

Note We recommend you use the default bandwidth value. EIGRP also uses the bandwidth parameter.

- MTU—The smallest maximum transmission unit value along the route to the destination.

- Hop count—The number of hops or routers that the route passes through to the destination. This metric is not directly used in the DUAL computation.
- Reliability—An indication of the reliability of the links to the destination.
- Load—An indication of how much traffic is on the links to the destination.

By default, EIGRP uses the bandwidth and delay metrics to calculate the distance to the destination. You can modify the metric weights to include the other metrics in the calculation.

External Route Metrics

External routes are routes that occur between neighbors in different EIGRP autonomous systems. These routes have the following metrics:

- Next hop—The IP address of the next-hop router.
- Router ID—The router ID of the router that redistributed this route into EIGRP.
- AS Number—The autonomous system number of the destination.
- Protocol ID—A code that represents the routing protocol that learned the destination route.
- Tag—An arbitrary tag that can be used for route maps.
- Metric—The route metric for this route from the external routing protocol.

EIGRP and the Unicast RIB

EIGRP adds all learned routes to the EIGRP topology table and the unicast RIB. When a topology change occurs, EIGRP uses these routes to search for a feasible successor. EIGRP also listens for notifications from the unicast RIB for changes in any routes redistributed to EIGRP from another routing protocol.

Advanced EIGRP

You can use the advanced features of EIGRP to optimize your EIGRP configuration. This section includes the following topics:

- Address Families, page 1-4
- Authentication, page 1-5
- Stub Routers, page 1-5
- Route Summarization, page 1-6
- Route Redistribution, page 1-6
- Load Balancing, page 1-6
- Split Horizon, page 1-6
- BFD, page 1-7
- Virtualization Support, page 1-7

Address Families

EIGRP supports the IPv4 address famil.

Address family configuration mode includes the following EIGRP features:

- Authentication
- AS number
- Default route
- Metrics
- Distance
- Graceful restart
- Logging
- Load balancing
- Redistribution
- Router ID
- Stub router
- Timers

You cannot configure the same feature in more than one configuration mode. For example, if you configure the default metric in router configuration mode, you cannot configure the default metric in address family mode.

Authentication

You can configure authentication on EIGRP messages to prevent unauthorized or invalid routing updates in your network. EIGRP authentication supports MD5 authentication digest.

You can configure the EIGRP authentication per virtual routing and forwarding (VRF) instance or interface using key-chain management for the authentication keys. Key-chain management allows you to control changes to the authentication keys used by MD5 authentication digest. See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0,* for more details about creating key-chains.

For MD5 authentication, you configure a password that is shared at the local router and all remote EIGRP neighbors. When an EIGRP message is created, Cisco NX-OS creates an MD5 one-way message digest based on the message itself and the encrypted password and sends this digest along with the EIGRP message. The receiving EIGRP neighbor validates the digest using the same encrypted password. If the message has not changed, the calculation is identical and the EIGRP message is considered valid.

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

Stub Routers

You can use the EIGRP stub routing feature to improve network stability, reduce resource usage, and simplify stub router configuration. Stub routers connect to the EIGRP network through a remote router. See the "Stub Routing" section on page 1-7.

When using EIGRP stub routing, you need to configure the distribution and remote routers to use EIGRP and configure only the remote router as a stub. EIGRP stub routing does not automatically enable summarization on the distribution router. In most cases, you need to configure summarization on the distribution routers.

Without EIGRP stub routing, even after the routes that are sent from the distribution router to the remote router have been filtered or summarized, a problem might occur. For example, if a route is lost somewhere in the corporate network, EIGRP could send a query to the distribution router. The distribution router could then send a query to the remote router even if routes are summarized. If a problem communicating over the WAN link between the distribution router and the remote router occurs, EIGRP could get stuck in active condition and cause instability elsewhere in the network. EIGRP stub routing allows you to prevent queries to the remote router.

Route Summarization

You can configure a summary aggregate address for a specified interface. Route summarization 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 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.

If more specific routes are in the routing table, EIGRP advertises the summary address from the interface with a metric equal to the minimum metric of the more specific routes.

Note EIGRP does not support automatic route summarization.

Route Redistribution

You can use EIGRP to redistribute direct routes, static routes, routes learned by other EIGRP autonomous systems, or routes from other protocols. You configure route map with the redistribution to control which routes are passed into EIGRP. A route map allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. See Chapter 1, “Configuring Route Policy Manager.”

You also configure the default metric that is used for all imported routes into EIGRP.

Load Balancing

You can use load balancing to allow a router to distribute traffic over all the router network ports that are the same distance from the destination address. Load balancing increases the utilization of network segments, which increases effective network bandwidth.

Cisco NX-OS supports the Equal Cost Multiple Paths (ECMP) feature with up to 64 equal-cost paths in the EIGRP route table and the unicast RIB. You can configure EIGRP to load balance traffic across some or all of those paths.

Note EIGRP in Cisco NX-OS does not support unequal cost load balancing.

Split Horizon

You can use split horizon to ensure that EIGRP never advertises a route out of the interface where it was learned.

Split horizon is a method that controls the sending of EIGRP update and query packets. When you enable split horizon on an interface, Cisco NX-OS does not send update and query packets for destinations that were learned from this interface. Controlling update and query packets in this manner reduces the possibility of routing loops.

Split horizon with poison reverse configures EIGRP to advertise a learned route as unreachable back through that the interface that EIGRP learned the route from.

EIGRP uses split horizon or split horizon with poison reverse in the following scenarios:

- Exchanging topology tables for the first time between two routers in startup mode.
- Advertising a topology table change.
- Sending a query message.

By default, the split horizon feature is enabled on all interfaces.

BFD

This feature supports bidirectional forwarding detection (BFD). BFD is a detection protocol designed to provide fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules. See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x* for more information.

Virtualization Support

Cisco NX-OS supports multiple instances of the EIGRP protocol that runs on the same system. EIGRP supports Virtual Routing and Forwarding instances (VRFs).

By default, every instance uses the same system router ID. You can optionally configure a unique router ID for each instance.

Licensing Requirements for EIGRP

The following table shows the licensing requirements for this feature:

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

Prerequisites for EIGRP

EIGRP has the following prerequisites:

You must enable the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

Guidelines and Limitations

EIGRP has the following configuration guidelines and limitations:

- A metric configuration (either through the default-metric configuration option or through a route map) is required for redistribution from any other protocol, connected routes, or static routes (see Chapter 1, "Configuring Route Policy Manager").

- For graceful restart, an NSF-aware router must be up and completely converged with the network before it can assist an NSF-capable router in a graceful restart operation.
- For graceful restart, neighboring switches participating in the graceful restart must be NSF-aware or NSF-capable.
- Cisco NX-OS EIGRP is compatible with EIGRP in the Cisco IOS software.
- Do not change the metric weights without a good reason. If you change the metric weights, you must apply the change to all EIGRP routers in the same autonomous system.
- Consider using stubs for larger networks.
- Avoid redistribution between different EIGRP autonomous systems because the EIGRP vector metric will not be preserved.
- The **no ip next-hop-self** command does not guarantee reachability of the next hop.
- The **ip passive-interface eigrp** command suppresses neighbors from forming.
- Cisco NX-OS does not support IGRP or connecting IGRP and EIGRP clouds.
- Autosummarization is not enabled by default.
- Cisco NX-OS supports only IP.

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

Table 1-1 lists the default settings for EIGRP parameters.

Table 1-1 Default EIGRP Parameters

Parameters	Default
Administrative distance	• Internal routes—90 • External routes—170
Bandwidth percent	50 percent
Default metric for redistributed routes	• bandwidth—100000 Kb/s • delay—100 (10 microsecond units) • reliability—255 • loading—1 • MTU—1500
EIGRP feature	Disabled
Hello interval	5 seconds
Hold time	15 seconds
Equal-cost paths	8
Metric weights	1 0 1 0 0
Next-hop address advertised	IP address of local interface

Table 1-1 Default EIGRP Parameters (continued)

Parameters	Default
NSF convergence time	120
NSF route-hold time	240
NSF signal time	20
Redistribution	Disabled
Split horizon	Enabled

Configuring Basic EIGRP

This section includes the following topics:

- Enabling the EIGRP Feature, page 1-9
- Creating an EIGRP Instance, page 1-10
- Restarting an EIGRP Instance, page 1-12
- Shutting Down an EIGRP Instance, page 1-12
- Shutting Down EIGRP on an Interface, page 1-13

Enabling the EIGRP Feature

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

SUMMARY STEPS

1. **configure terminal**
2. **feature eigrp**
3. (Optional) **show feature**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`feature eigrp`**
	
	`Example:`
	`switch(config)# feature eigrp`
		Enables the EIGRP feature.

	Command	Purpose
Step 3	**show feature** **Example:** `switch(config)# show feature`	(Optional) Displays information about enabled features.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

Use the **no feature eigrp** command to disable the EIGRP feature and remove all associated configuration.

Command	Purpose
no feature eigrp **Example:** `switch(config)# no feature eigrp`	Disables the EIGRP feature and removes all associated configuration.

Creating an EIGRP Instance

You can create an EIGRP instance and associate an interface with that instance. You assign a unique autonomous system number for this EIGRP process (see the "Autonomous Systems" section on page 1-5). Routes are not advertised or accepted from other autonomous systems unless you enable route redistribution.

BEFORE YOU BEGIN

Ensure that you have enabled the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

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

SUMMARY STEPS

1. If you configure an instance tag that does not qualify as an AS number, you must configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.
2. **configure terminal**
3. **router eigrp** *instance-tag*
4. (Optional) **log-adjacency-changes**
5. (Optional) **log-neighbor-warnings** [***seconds***]
6. **interface** *interface-type slot/port*
7. **no switchport**
8. **ip router eigrp** *instance-tag*
9. **show ip eigrp interfaces**
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router eigrp`** *`instance-tag`*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**`log-adjacency-changes`**
	Example:
	`switch(config-router)#`
	`log-adjacency-changes`
		(Optional). Generates a system message whenever an adjacency changes state. This command is enabled by default.

Step 4	**`log-neighbor-warnings`** `[`*`seconds`*`]`
	Example:
	`switch(config-router)#`
	`log-neighbor-warnings`
		(Optional) Generates a system message whenever a neighbor warning occurs. You can configure the time between warning messages, from 1 to 65535, in seconds. The default is 10 seconds. This command is enabled by default.

Step 5	**`interface`** *`interface-type slot/port`*
	Example:
	`switch(config-router)# interface`
	`ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode. Use **?** to determine the slot and port ranges.
		Note If this is a 10G breakout port, the *slot*/*port* syntax is *slot*/*QSFP-module*/*port*.

Step 6	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 7	**`ip router eigrp`** *`instance-tag`*
	Example:
	`switch(config-if)# ip router eigrp Test1`
		Associates this interface with the configured EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Step 8	**`show ip eigrp interfaces`**
	Example:
	`switch(config-if)# show ip eigrp`
	`interfaces`
		Displays information about EIGRP interfaces.

Step 9	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config`
	`startup-config`
		(Optional) Saves this configuration change.

Use the **no router eigrp** command to remove the EIGRP process and the associated configuration.

Command	Purpose
no router eigrp *instance-tag* **Example:** `switch(config)# no router eigrp Test1`	Deletes the EIGRP process and all associated configuration.

Note You should also remove any EIGRP commands configured in interface mode if you remove the EIGRP process.

This example shows how to create an EIGRP process and configure an interface for EIGRP:

```
switch# configure terminal
switch(config-router)# router eigrp Test1
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# autonomous-system 1
switch(config-router-af)# exit
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ipv6 router eigrp Test1
switch(config-if)# no shutdown
switch(config-if)# copy running-config startup-config
```

For more information about other EIGRP parameters, see the "Configuring Advanced EIGRP" section on page 1-13.

Restarting an EIGRP Instance

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

To restart an EIGRP instance and remove all associated neighbors, use the following commands:

Command	Purpose
flush-routes **Example:** `switch(config)# flush-routes`	(Optional) Flushes all EIGRP routes in the unicast RIB when this EIGRP instance restarts.
restart eigrp *instance-tag* **Example:** `switch(config)# restart eigrp Test1`	Restarts the EIGRP instance and removes all neighbors. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Shutting Down an EIGRP Instance

You can gracefully shut down an EIGRP instance. This action emoves all routes and adjacencies but preserves the EIGRP configuration.

To disable an EIGRP instance, use the following command in address family mode:

Command	Purpose
`switch(config-router-af)# shutdown` **Example:** `switch(config-router-af)# shutdown`	Disables this instance of EIGRP. The EIGRP router configuration remains.

Configuring a Passive Interface for EIGRP

You can configure a passive interface for EIGRP. A passive interface does not participate in EIGRP adjacency but the network address for the interfacee remains in the EIGRP topology table.

To configure a passive interface for EIGRP, use the following command in interface configuration mode:

Command	Purpose
ip passive-interface eigrp *instance-tag*	Suppresses EIGRP hellos, which prevents neighbors from forming and sending routing updates on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Shutting Down EIGRP on an Interface

You can gracefully shut down EIGRP on an interface. This action removes all adjacencies and stops EIGRP traffic on this interface but preserves the EIGRP configuration.

To disable EIGRP on an interface, use the following command in interface configuration mode:

Command	Purpose
switch(config-if)# **ip eigrp** *instance-tag* **shutdown** **Example:** `switch(config-router)# ip eigrp Test1 shutdown`	Disables EIGRP on this interface. The EIGRP interface configuration remains. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Configuring Advanced EIGRP

This section includes the following topics:

- Configuring Authentication in EIGRP, page 1-14
- Configuring EIGRP Stub Routing, page 1-16
- Configuring a Summary Address for EIGRP, page 1-17
- Redistributing Routes into EIGRP, page 1-17
- Limiting the Number of Redistributed Routes, page 1-19
- Configuring Load Balancing in EIGRP, page 1-21

- Adjusting the Interval Between Hello Packets and the Hold Time, page 1-22
- Disabling Split Horizon, page 1-23
- Tuning EIGRP, page 1-23

Configuring Authentication in EIGRP

You can configure authentication between neighbors for EIGRP. See the "Authentication" section on page 1-5.

You can configure EIGRP authentication for the EIGRP process or for individual interfaces. Interface EIGRP authentication configuration overrides the EIGRP process-level authentication configuration.

BEFORE YOU BEGIN

Ensure that you have enabled the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

Ensure that all neighbors for an EIGRP process share the same authentication configuration, including the shared authentication key.

Create the key-chain for this authentication configuration. See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*.

SUMMARY STEPS

1. **configure terminal**
2. **router eigrp** *instance-tag*
3. **address-family ipv4 unicast**
4. **authentication key-chain** *key-chain*
5. **authentication mode md5**
6. **interface** *interface-type slot/port*
7. **no switchport**
8. **ip router eigrp** *instance-tag*
9. **ip authentication key-chain eigrp** *instance-tag key-chain*
10. **ip authentication mode eigrp** *instance-tag* **md5**
11. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router eigrp`** *`instance-tag`*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**`address-family`** `{`**`ipv4 unicast`**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters the address-family configuration mode. This command is optional for IPv4.

Step 4	**`authentication key-chain`** *`key-chain`*
	Example:
	`switch(config-router-af)# authentication key-chain routeKeys`
		Associates a key chain with this EIGRP process for this VRF. The key chain can be any case-sensitive, alphanumeric string up to 20 characters.

Step 5	**`authentication mode md5`**
	Example:
	`switch(config-router-af)# authentication mode md5`
		Configures MD5 message digest authentication mode for this VRF.

Step 6	**`interface`** *`interface-type slot/port`*
	Example:
	`switch(config-router-af) interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode. Use **?** to find the supported interfaces.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 7	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 8	`{`**`ip router eigrp`** *`instance-tag`*
	Example:
	`switch(config-if)# ip router eigrp Test1`
		Associates this interface with the configured EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

	Command	Purpose
Step 9	{**ip authentication key-chain eigrp** *instance-tag key-chain* **Example:** `switch(config-if)# ip authentication key-chain eigrp Test1 routeKeys`	Associates a key chain with this EIGRP process for this interface. This configuration overrides the authentication configuration set in the router VRF mode. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 10	{**ip authentication mode eigrp** *instance-tag* **md5** **Example:** `switch(config-if)# ip authentication mode eigrp Test1 md5`	Configures the MD5 message digest authentication mode for this interface. This configuration overrides the authentication configuration set in the router VRF mode. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 11	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure MD5 message digest authentication for EIGRP over Ethernet interface 1/2:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# exit
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip router eigrp Test1
switch(config-if)# ip authentication key-chain eigrp Test1 routeKeys
switch(config-if)# ip authentication mode eigrp Test1 md5
switch(config-if)# copy running-config startup-config
```

Configuring EIGRP Stub Routing

To configure a router for EIGRP stub routing, use the following command in address-family configuration mode:

Command	Purpose
switch(config-router-af)# **stub** [**direct** | **receive-only** | **redistributed** [**direct**] **leak-map** *map-name*] **Example:** `switch(config-router-af)# eigrp stub redistributed`	Configures a remote router as an EIGRP stub router. The map name can be any case-sensitive, alphanumeric string up to 20 characters.

This example shows how to configure a stub router to advertise directly connected and redistributed routes:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# stub direct redistributed
switch(config-router-af)# copy running-config startup-config
```

Use the **show ip eigrp neighbor detail** command to verify that a router has been configured as a stub router. The last line of the output shows the stub status of the remote or spoke router. This example shows the output from the **show ip eigrp neighbor detail** command:

```
Router# show ip eigrp neighbor detail
IP-EIGRP neighbors for process 201
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
0   10.1.1.2                Se3/1         11 00:00:59    1  4500  0  7
   Version 12.1/1.2, Retrans: 2, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
```

Configuring a Summary Address for EIGRP

You can configure a summary aggregate address for a specified interface. If any more specific routes are in the routing table, EIGRP will advertise the summary address out the interface with a metric equal to the minimum of all more specific routes. See the "Route Summarization" section on page 1-6.

To configure a summary aggregate address, use the following command in interface configuration mode:

Command	Purpose

`switch(config-if)# {`**`ip summary-address eigrp`** *`instance-tag ip-prefix/length`* `[`*`distance`* `|` **`leak-map`** *`map-name`*`]`
Example:
`switch(config-if)# ip summary-address eigrp Test1 192.0.2.0/8`
	Configures a summary aggregate address as either an IP address and network mask, or an IP prefix/length. The instance tag and map name can be any case-sensitive, alphanumeric string up to 20 characters.
	You can optionally configure the administrative distance for this aggregate address. The default administrative distance is 5 for aggregate addresses.

This example causes EIGRP to summarize network 192.0.2.0 out Ethernet 1/2 only:

```
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip summary-address eigrp Test1 192.0.2.0 255.255.255.0
```

Redistributing Routes into EIGRP

You can redistribute routes in EIGRP from other routing protocols.

Note Redistribution does not work if the access list is used as a **match** option in **route-maps.**

BEFORE YOU BEGIN

Ensure that you have enabled the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

You must configure the metric (either through the default-metric configuration option or through a route map) for routes redistributed from any other protocol.

You must create a route map to control the types of routes that are redistributed into EIGRP. See Chapter 1, "Configuring Route Policy Manager."

SUMMARY STEPS

1. **configure terminal**
2. **router eigrp** *instance-tag*
3. **address-family ipv4 unicast**
4. **redistribute** {**bgp** *as* | {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **direct** | **static**} **route-map** *name*
5. **default-metric** *bandwidth delay reliability loading mtu*
6. **show ip eigrp route-map statistics redistribute**
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router eigrp** *instance-tag*
	
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**address-family** {**ipv4 unicast**
	
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters the address-family configuration mode. This command is optional for IPv4.

	Command	Purpose
Step 4	**redistribute** {**bgp** *as*| {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **direct** | **static**} **route-map** *name* **Example:** `switch(config-router-af)# redistribute bgp 100 route-map BGPFilter`	Injects routes from one routing domain into EIGRP. The instance tag and map name can be any case-sensitive, alphanumeric string up to 20 characters.
Step 5	**default-metric** *bandwidth delay reliability loading mtu* **Example**: `switch(config-router-af)# default-metric 500000 30 200 1 1500`	Sets the metrics assigned to routes learned through route redistribution. The default values are as follows: • bandwidth—100000 Kb/s • delay—100 (10 microsecond units) • reliability—255 • loading—1 • MTU—1492
Step 6	**show** {**ip eigrp route-map statistics redistribute** **Example**: `switch(config-router-af)# show ip eigrp route-map statistics redistribute bgp`	Displays information about EIGRP route map statistics.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to redistribute BGP into EIGRP for IPv4:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# redistribute bgp 100 route-map BGPFilter
switch(config-router)# default-metric 500000 30 200 1 1500
switch(config-router)# copy running-config startup-config
```

Limiting the Number of Redistributed Routes

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

- Fixed limit—Logs a message when EIGRP reaches the configured maximum. EIGRP does not accept any more redistributed routes. You can optionally configure a threshold percentage of the maximum where EIGRP will log a warning when that threshold is passed.
- Warning only—Logs a warning only when EIGRP reaches the maximum. EIGRP continues to accept redistributed routes.
- Withdraw—Start the timeout period when EIGRP reaches the maximum. After the timeout period, EIGRP 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, EIGRP

withdraws all redistributed routes. You must clear this condition before EIGRP accepts more redistributed routes.
You can optionally configure the timeout period.

BEFORE YOU BEGIN

Ensure that you have enabled the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

SUMMARY STEPS

1. **configure terminal**
2. **router eigrp** *instance-tag*
3. **redistribute** {**bgp** *id* | **direct** | **eigrp** *id* | **ospf** *id* | **rip** *id* | **static**} **route-map** *map-name*
4. **redistribute maximum-prefix** *max* [*threshold*] [**warning-only** | **withdraw** [*num-retries timeout*]]
5. (Optional) **show running-config eigrp**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router eigrp`** *`instance-tag`*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP instance with the configured instance tag.

Step 3	**`redistribute`** {**`bgp`** *`id`* | **`direct`** | **`eigrp`** *`id`* | **`ospf`** *`id`* | **`rip`** *`id`* | **`static`**} **`route-map`** *`map-name`*
	Example:
	`switch(config-router)# redistribute bgp route-map FilterExternalBGP`
		Redistributes the selected protocol into EIGRP through the configured route map.

	Command	Purpose

Step 4	**`redistribute maximum-prefix`** *`max`* `[`*`threshold`*`] [`**`warning-only`** `|` **`withdraw`** `[`*`num-retries timeout`*`]]`
	
	`Example:`
	`switch(config-router)# redistribute maximum-prefix 1000 75 warning-only`
		Specifies a maximum number of prefixes that EIGRP will distribute. The range is from 0 to 65536. Optionally specifies the following:
		• *threshold*—Percent of maximum prefixes that will trigger a warning message.
		• **warning-only**—Logs an warning message when the maximum number of prefixes is exceeded.
		• **withdraw**—Withdraws all redistributed routes. Optionally tries to retrieve the redistributed routes. The *num-retries* range is from 1 to 12. The *timeout* is from 60 to 600 seconds. The default is 300 seconds. Use **clear ip eigrp redistribution** if all routes are withdrawn.

Step 5	**`show running-config eigrp`**
	
	`Example:`
	`switch(config-router)# show running-config eigrp`
		(Optional) Displays the EIGRP configuration.

Step 6	**`copy running-config startup-config`**
	
	`Example:`
	`switch(config-router)# copy running-config startup-config`
		(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# redistribute bgp route-map FilterExternalBGP
switch(config-router)# redistribute maximum-prefix 1000 75
```

Configuring Load Balancing in EIGRP

You can configure load balancing in EIGRP. You can configure the number of Equal Cost Multiple Path (ECMP) routes using the maximum paths option. See the "Configuring Load Balancing in EIGRP" section on page 1-21.

BEFORE YOU BEGIN

Ensure that you have enabled the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

SUMMARY STEPS

1. **configure terminal**
2. **router eigrp** *instance-tag*
3. **address-family ipv4 unicast**
4. **maximum-paths** *num-paths*

5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router eigrp`** *`instance-tag`*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

Step 3	**`address-family`** `{`**`ipv4 unicast`**
	Example:
	`switch(config-router)# address-family ipv4 unicast`
	`switch(config-router-af)#`
		Enters the address-family configuration mode. This command is optional for IPv4.

Step 4	**`maximum-paths`** *`num-paths`*
	Example:
	`switch(config-router-af)# maximum-paths 5`
		Sets the number of equal cost paths that EIGRP will accept in the route table. The range is from 1 to 64. The default is 8.

Step 5	**`copy running-config startup-config`**
	Example:
	`switch(config-router-af)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to configure equal cost load balancing for EIGRP over IPv4 with a maximum of six equal cost paths:

```
switch# configure terminal
switch(config)# router eigrp Test1
switch(config-router)# maximum-paths 6
switch(config-router)# copy running-config startup-config
```

Adjusting the Interval Between Hello Packets and the Hold Time

You can adjust the interval between hello messages and the hold time.

By default, hello messages are sent every 5 seconds. The hold time is advertised in hello messages and indicates to neighbors the length of time that they should consider the sender valid. The default hold time is three times the hello interval, or 15 seconds.

To change the interval between hello packets, use the following command in interface configuration mode:

Command	Purpose

`switch(config-if)# {ip hello-interval eigrp instance-tag seconds`
Example:
`switch(config-if)# ip hello-interval eigrp Test1 30`
	Configures the hello interval for an EIGRP routing process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The range is from 1 to 65535 seconds. The default is 5.

On very congested and large networks, the default hold time might not be sufficient time for all routers to receive hello packets from their neighbors. In this case, you might want to increase the hold time.

To change the hold time, use the following command in interface configuration mode:

Command	Purpose

`switch(config-if)# {ip hold-time eigrp instance-tag seconds`
Example:
`switch(config-if)# ip hold-time eigrp Test1 30`
	Configures the hold time for an EIGRP routing process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The range is from 1 to 65535.

Use the **show ip eigrp interface detail** command to verify timer configuration.

Disabling Split Horizon

You can use split horizon to block route information from being advertised by a router out of any interface from which that information originated. Split horizon usually optimizes communications among multiple routing switches, particularly when links are broken.

By default, split horizon is enabled on all interfaces.

To disable split horizon, use the following command in interface configuration mode:

Command	Purpose

`switch(config-if)# no {ip split-horizon eigrp instance-tag`
Example:
`switch(config-if)# no ip split-horizon eigrp Test1`
	Disables split horizon.

Tuning EIGRP

You can configure optional parameters to tune EIGRP for your network.

You can configure the following optional parameters in address-family configuration mode:

Command	Purpose
default-information originate [**always** | **route-map** *map-name*] **Example:** `switch(config-router-af)# default-information originate always`	Originates or accepts the default route with prefix 0.0.0.0/0. When a route map is supplied, the default route is originated only when the route map yields a true condition. The map name can be any case-sensitive, alphanumeric string up to 20 characters.
distance *internal external* **Example**: `switch(config-router-af)# distance 25 100`	Configures the administrative distance for this EIGRP process. The range is from 1 to 255. The internal value sets the distance for routes learned from within the same autonomous system (the default value is 90). The external value sets the distance for routes learned from an external autonomous system (the default value is 170).
metric maximum-hops *hop-count* **Example:** `switch(config-router-af)# metric maximum-hops 70`	Sets maximum allowed hops for an advertised route. Routes over this maximum are advertised as unreachable. The range is from 1 to 255. The default is 100.
metric weights *tos k1 k2 k3 k4 k5* **Example:** `switch(config-router-af)# metric weights 0 1 3 2 1 0`	Adjusts the EIGRP metric or K value. EIGRP uses the following formula to determine the total metric to the network: metric = [k1*bandwidth + (k2*bandwidth)/(256 – load) + k3*delay] * [k5/(reliability + k4)] Default values and ranges are as follows: • TOS—0. The range is from 0 to 8. • k1—1. The range is from 0 to 255. • k2—0. The range is from 0 to 255. • k3—1. The range is from 0 to 255. • k4—0. The range is from 0 to 255. • k5—0. The range is from 0 to 255.

timers active-time {*time-limit* | **disabled**}
Example:
`switch(config-router-af)# timers active-time 200.`
	Sets the time the router waits in minutes (after sending a query) before declaring the route to be stuck in the active (SIA) state. The range is from 1 to 65535. The default is 3.

You can configure the following optional parameters in interface configuration mode:

Command	Purpose
{**ip bandwidth eigrp** *instance-tag bandwidth* **Example:** switch(config-if)# ip bandwidth eigrp Test1 30000	Configures the bandwidth metric for EIGRP on an interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The bandwidth range is from 1 to 2,560,000,000 Kb/s.
{**ip bandwidth-percent eigrp** *instance-tag percent* **Example:** switch(config-if)# ip bandwidth-percent eigrp Test1 30	Configures the percentage of bandwidth that EIGRP might use on an interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The percent range is from 0 to 100. The default is 50.
no ip delay eigrp *instance-tag delay* **Example:** switch(config-if)# ip delay eigrp Test1 100	Configures the delay metric for EIGRP on an interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters. The delay range is from 1 to 16777215 (in tens of microseconds).
{**ip distribute-list eigrp** *instance-tag* {**prefix-list** *name*| **route-map** name} {**in** | **out**} **Example:** switch(config-if)# ip distribute-list eigrp Test1 route-map EigrpTest in	Configures the route filtering policy for EIGRP on this interface. The instance tag, prefix list name, and route map name can be any case-sensitive, alphanumeric string up to 20 characters.
no {**ip next-hop-self eigrp** *instance-tag* **Example:** switch(config-if)# ip next-hop-self eigrp Test1	Configures EIGRP to use the received next-hop address rather than the address for this interface. The default is to use the IP address of this interface for the next-hop address. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
{**ip offset-list eigrp** *instance-tag* {**prefix-list** *name*| **route-map** name} {**in** | **out**} *offset* **Example:** switch(config-if)# ip offfset-list eigrp Test1 prefix-list EigrpList in	Adds an offset to incoming and outgoing metrics to routes learned by EIGRP. The instance tag, prefix list name, and route map name can be any case-sensitive, alphanumeric string up to 20 characters.
{**ip passive-interface eigrp** *instance-tag* **Example:** switch(config-if)# ip passive-interface eigrp Test1	Suppresses EIGRP hellos, which prevents neighbors from forming and sending routing updates on an EIGRP interface. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.

Configuring Virtualization for EIGRP

You can create multiple VRFs and use the same or multiple EIGRP processes in each VRF. You assign an 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 other configuration for that interface.

BEFORE YOU BEGIN

Ensure that you have enabled the EIGRP feature (see the "Enabling the EIGRP Feature" section on page 1-9).

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **router eigrp** *instance-tag*
4. **interface ethernet** *slot/port*
5. **no switchport**
6. **vrf member** *vrf-name*
7. **ip router eigrp** *instance-tag*
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`vrf context`** *`vrf-name`*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode. The VRN name can be any case-sensitive, alphanumeric string up to 20 characters.

Step 3	**`router eigrp`** *`instance-tag`*
	Example:
	`switch(config)# router eigrp Test1`
	`switch(config-router)#`
		Creates a new EIGRP process with the configured instance tag. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
		If you configure an *instance-tag* that does not qualify as an AS number, you must use the **autonomous-system** command to configure the AS number explicitly or this EIGRP instance will remain in the shutdown state.

	Command	Purpose

Step 4	**interface ethernet** *slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode. Use **?** to find the slot and port ranges.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 5	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.
Step 6	**vrf member** *vrf-name* **Example:** `switch(config-if)# vrf member RemoteOfficeVRF`	Adds this interface to a VRF. The VRF name can be any case-sensitive, alphanumeric string up to 20 characters.
Step 7	{**ip router eigrp** *instance-tag* **Example**: `switch(config-if)# ip router eigrp Test1`	Adds this interface to the EIGRP process. The instance tag can be any case-sensitive, alphanumeric string up to 20 characters.
Step 8	**copy running-config startup-config** **Example:** `switch(config-if)# copy running-config startup-config`	(Optional) Saves this configuration change.

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-vrf)# router eigrp Test1
switch(config-router)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip router eigrp Test1
switch(config-if)# vrf member NewVRF
switch(config-if)# copy running-config startup-config
```

Verifying the EIGRP Configuration

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

Command	Purpose
show ip eigrp [*instance-tag*]	Displays a summary of the configured EIGRP processes.
show ip eigrp [*instance-tag*] **interfaces** [*type number*] [**brief**] [**detail**]	Displays information about all configured EIGRP interfaces.
show ip eigrp *instance-tag* **neighbors** [*type number*]	Displays information about all the EIGRP neighbors. Use this command to verify the EIGRP neighbor configuration.

Command	Purpose
show ip eigrp [*instance-tag*] **route** [*ip-prefix/length*] [**active**] [**all-links**] [**detail-links**] [**pending**] [**summary**] [**zero-successors**] [**vrf** *vrf-name*]	Displays information about all the EIGRP routes.
show ip eigrp [*instance-tag*] **topology** [*ip-prefix/length*] [**active**] [**all-links**] [**detail-links**] [**pending**] [**summary**] [**zero-successors**] [**vrf** *vrf-name*]	Displays information about the EIGRP topology table.
show running-configuration eigrp	Displays the current running EIGRP configuration.

Displaying EIGRP Statistics

To display EIGRP statistics, use the following commands:

Command	Purpose
show ip eigrp [*instance-tag*] **accounting** [**vrf** *vrf-name*]	Displays accounting statistics for EIGRP.
show ip eigrp [*instance-tag*] **route-map statistics redistribute**	Displays redistribution statistics for EIGRP.
show ip eigrp [*instance-tag*] **traffic** [**vrf** *vrf-name*]	Displays traffic statistics for EIGRP.

Configuration Examples for EIGRP

This example shows how to configure EIGRP:

```
feature eigrp
interface ethernet 1/2
 no switchport
 ip address 192.0.2.55/24
 ip router eigrp Test1
  no shutdown
router eigrp Test1
  router-id 192.0.2.1
```

Related Topics

See Chapter 1, “Configuring Route Policy Manager” for more information on route maps.

Additional References

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

- Related Documents, page 1-29
- MIBs, page 1-29

Related Documents

Related Topic	Document Title
EIGRP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*
http://www.cisco.com/warp/public/103/1.html	*Introduction to EIGRP Tech Note*
http://www.cisco.com/en/US/tech/tk365/technologies_q_and_a_item09186a008012dac4.shtml	EIGRP Frequently Asked Questions

MIBs

MIBs	MIBs Link
CISCO-EIGRP-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

CHAPTER 1

Configuring Basic BGP

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

This chapter includes the following sections:

Information 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 switches.

BGP uses a path-vector routing algorithm to exchange routing information between BGP-enabled networking switches 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 additional 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" section on page 1-3 for more information.

BGP also supports load balancing or equal-cost multipath (ECMP). See the "Load Sharing and Multipath" section on page 1-6 for more information.

This section includes the following topics:

- BGP Autonomous Systems, page 1-2
- Administrative Distance, page 1-2
- BGP Peers, page 1-3
- BGP Router Identifier, page 1-3
- BGP Path Selection, page 1-4
- BGP and the Unicast RIB, page 1-7
- BGP Virtualization, page 1-7

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. For more information, see the "Autonomous Systems" section on page 1-5.

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 or 4-byte AS numbers. Cisco NX-OS displays 4-byte AS numbers in plain-text notation (that is, as 32-bit integers). You can configure 4-byte AS numbers as either plain-text notation (for example, 1 to 4294967295), or AS.dot notation (for example, 1.0). For more information, see the "Autonomous Systems" section on page 1-5.

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 Table 1-1.

Table 1-1 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	200	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.

For more information, see the "Administrative Distance" section on page 1-7.

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 or 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.

Dynamis AS Numbers for Prefix 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 Chapter 1, "Configuring Advanced BGP," 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. See Chapter 1, "Configuring Advanced BGP," for more information on templates.

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, then 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 Path Selection

Although BGP might receive advertisements for the same route from multiple sources, BGP selects only one path as the best path. BGP puts the selected path in the IP routing table and propagates the path to its peers.

The best-path algorithm runs each time that a path is added or withdrawn for a given network. The best-path algorithm also runs if you change the BGP configuration. BGP selects the best path from the set of valid paths available for a given network.

Cisco NX-OS implements the BGP best-path algorithm in the following steps:

Step 1 Compares two paths to determine which is better (see the "Step 1—Comparing Pairs of Paths" section on page 1-4).

Step 2 Iterates over all paths and determines in which order to compare the paths to select the overall best path (see the "Step 2—Determining the Order of Comparisons" section on page 1-6).

Step 3 Determines whether the old and new best paths differ enough so that the new best path should be used (see the "Step 3—Determining the Best-Path Change Suppression" section on page 1-6).

Note The order of comparison determined in Part 2 is important. Consider the case where you have three paths, A, B, and C. When Cisco NX-OS compares A and B, it chooses A. When Cisco NX-OS compares B and C, it chooses B. But when Cisco NX-OS compares A and C, it might not choose A because some BGP metrics apply only among paths from the same neighboring autonomous system and not among all paths.

The path selection uses the the BGP AS-path attribute. The AS-path attribute includes the list of autonomous system numbers (AS numbers) traversed in the advertised path. If you subdivide your BGP autonomous system into a collection or confederation of autonomous systems, the AS path contains confederation segments that list these locally defined autonomous systems.

Step 1—Comparing Pairs of Paths

This first step in the BGP best-path algorithm compares two paths to determine which path is better. The following sequence describes the basic steps that Cisco NX-OS uses to compare two paths to determine the better path:

1. Cisco NX-OS chooses a valid path for comparison. (For example, a path that has an unreachable next hop is not valid.)
2. Cisco NX-OS chooses the path with the highest weight.
3. Cisco NX-OS chooses the path with the highest local preference.
4. If one of the paths is locally originated, Cisco NX-OS chooses that path.

5. Cisco NX-OS chooses the path with the shorter AS path.

Note When calculating the length of the AS path, Cisco NX-OS ignores confederation segments, and counts AS sets as 1. See the "AS Confederations" section on page 1-4 for more information.

6. Cisco NX-OS chooses the path with the lower origin. Interior Gateway Protocol (IGP) is considered lower than EGP.
7. Cisco NX-OS chooses the path with the lower multi- exit discriminator (MED).

   You can configure a number of options that affect whether or not this step is performed. In general, Cisco NX-OS compares the MED of both paths if the paths were received from peers in the same autonomous system; otherwise, Cisco NX-OS skips the MED comparison.

   You can configure Cisco NX-OS to always perform the best-path algorithm MED comparison, regardless of the peer autonomous system in the paths. See the "Tuning the Best-Path Algorithm" section on page 1-9 for more information. Otherwise, Cisco NX-OS will perform a MED comparison that depends on the AS-path attributes of the two paths being compared:

   a. If a path has no AS path or the AS path starts with an AS_SET, then the path is internal, and Cisco NX-OS compares the MED to other internal paths.
   b. If the AS path starts with an AS_SEQUENCE, then the peer autonomous system is the first AS number in the sequence, and Cisco NX-OS compares the MED to other paths that have the same peer autonomous system.
   c. If the AS path contains only confederation segments or starts with confederation segments followed by an AS_SET, the path is internal and Cisco NX-OS compares the MED to other internal paths.
   d. If the AS path starts with confederation segments followed by an AS_SEQUENCE, then the peer autonomous system is the first AS number in the AS_SEQUENCE, and Cisco NX-OS compares the MED to other paths that have the same peer autonomous system.

   Note If Cisco NX-OS receives no MED attribute with the path, then Cisco NX-OS considers the MED to be 0 unless you configure the best-path algorithm to set a missing MED to the highest possible value. See the "Tuning the Best-Path Algorithm" section on page 1-9 for more information.

   e. If the nondeterministic MED comparison feature is enabled, the best path algorithm uses the Cisco IOS style of MED comparison. See the "Tuning the Best-Path Algorithm" section on page 1-9 for more information.
8. If one path is from an internal peer and the other path is from an external peer, then Cisco NX-OS chooses the path from the external peer.
9. If the paths have different IGP metrics to their next-hop addresses, then Cisco NX-OS chooses the path with the lower IGP metric.
10. Cisco NX-OS uses the path that was selected by the best-path algorithm the last time that it was run.

If all path parameters in Step 1 through Step 9 are the same, then you can configure the best-path algorithm to compare the router IDs. See the "Tuning the Best-Path Algorithm" section on page 1-9 for more information. If the path includes an originator attribute, then Cisco NX-OS uses that attribute as the router ID to compare to; otherwise, Cisco NX-OS uses the router ID of the peer that sent the path. If the paths have different router IDs, Cisco NX-OS chooses the path with the lower router ID.

Note When using the attribute originator as the router ID, it is possible that two paths have the same router ID. It is also possible to have two BGP sessions with the same peer router, and therefore you can receive two paths with the same router ID.

11. Cisco NX-OS selects the path with the shorter cluster length. If a path was not received with a cluster list attribute, the cluster length is 0.
12. Cisco NX-OS chooses the path received from the peer with the lower IP address. Locally generated paths (for example, redistributed paths) have a peer IP address of 0.

Note Paths that are equal after step 9 can be used for multipath if you configure multipath. See the "Load Sharing and Multipath" section on page 1-6 for more information.

Step 2—Determining the Order of Comparisons

The second step of the BGP best-path algorithm implementation is to determins the order in which Cisco NX-OS compares the paths:

1. Cisco NX-OS partitions the paths into groups. Within each group Cisco NX-OS compares the MED among all paths. Cisco NX-OS uses the same rules as in the "Step 1—Comparing Pairs of Paths" section on page 1-4 to determine whether MED can be compared between any two paths. Typically, this comparison results in one group being chosen for each neighbor autonomous system. If you configure the **bgp bestpath med always** command, then Cisco NX-OS chooses just one group that contains all the paths.
2. Cisco NX-OS determines the best path in each group by iterating through all paths in the group and keeping track of the best one so far. Cisco NX-OS compares each path with the temporary best path found so far and if the new path is better, it becomes the new temporary best path and Cisco NX-OS compares it with the next path in the group.
3. Cisco NX-OS forms a set of paths that contain the best path selected from each group in Step 2. Cisco NX-OS selects the overall best path from this set of paths by going through them as in Step 2.

Step 3—Determining the Best-Path Change Suppression

The next part of the implementation is to determine whether Cisco NX-OS will use the new best path or suppress the new best path. The router can continue to use the existing best path if the new one is identical to the old path (if the router ID is the same). Cisco NX-OS continues to use the existing best path to avoid route changes in the network.

You can turn off the suppression feature by configuring the best-path algorithm to compare the router IDs. See the "Tuning the Best-Path Algorithm" section on page 1-9 for more information. If you configure this feature, the new best path is always preferred to the existing one.

You cannot suppress the best-path change if any of the following conditions occur:

- The existing best path is no longer valid.
- Either the existing or new best paths were received from internal (or confederation) peers or were locally generated (for example, by redistribution).
- The paths were received from the same peer (the paths have the same router ID).
- The paths have different weights, local preferences, origins, or IGP metrics to their next-hop addresses.

- The paths have different MEDs.

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 instances (VRFs).

Licensing Requirements for Basic BGP

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	BGP requires a LAN Enterprise Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide.* **Note** Make sure the LAN Base Services license is installed on the switch to enable Layer 3 interfaces.

Prerequisites for BGP

BGP has the following prerequisites:

- You must enable the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).
- 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 BGP

BGP has the following configuration guidelines and limitations:

- The dynamic AS number prefix peer configuration the overrides individual AS number configuration inherited from a BGP template.
- If you configure a dynamic AS number for prefix peers in an AS confederation, BGP establishes sessions with only the AS numbers in the local confederation.
- BGP sessions created through a dynamic AS number prefix peer ignore any configured eBGP multihop time-to-live (TTL) value or a disabled check for directly connected peers.
- Configure a router ID for BGP to avoid automatic router ID changes and session flaps.
- Use the maximum-prefix configuration option per peer to restrict the number of routes received and system resources used.
- Configure the update-source to establish a session with 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.
- If you configure VRFs, enter the desired VRF (see Chapter 1, "Configuring Layer 3 Virtualization").

Default Settings

Table 1-2 lists the default settings for BGP parameters.

Table 1-2 Default BGP Parameters

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

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.

This section includes the following topics:

- Global Configuration Mode, page 1-8
- Address Family Configuration Mode, page 1-9
- Neighbor Configuration Mode, page 1-9
- Neighbor Address Family Configuration Mode, page 1-10

Global Configuration Mode

Use global configuration mode to create a BGP process and configure advanced features such as AS confederation and route dampening. For more information, see Chapter 1, "Configuring Advanced BGP."

This example shows how to enter router configuration mode:

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

BGP supports Virtual Routing and Forwarding (VRF). You can configure BGP within the appropriate VRF if you are using VRFs in your network. See the "Configuring Virtualization" section on page 1-35 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)#
```

Address Family Configuration Mode

You can optionally configure the address families that BGP supports. Use the **address-family** command in router configuration mode to configure features for an address family. Use the **address-family** command in neighbor configuration mode to configure the specific address family for the neighbor.

You must configure the address families if you are using route redistribution, address aggregation, load balancing, and other advanced features.

This example shows how to enter address family configuration mode from the router configuration mode:

```
switch(config)# router bgp 64496
switch(config-router)# address-family ipv6 unicast
switch(config-router-af)#
```

This example shows how to enter VRF address family configuration mode if you are using VRFs:

```
switch(config)# router bgp 64497
switch(config-router)# vrf vrf_A
switch(config-router-vrf)# address-family ipv6 unicast
switch(config-router-vrf-af)#
```

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.

This 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)#
```

This 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 neighbor address family configuration mode:

```
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 VRF neighbor address family configuration mode:

```
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 ipv6 unicast
switch(config-router-vrf-neighbor-af)#
```

Configuring Basic BGP

To configure a basic BGP, you need to 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.

This section includes the following topics:

- Enabling the BGP Feature, page 1-10
- Creating a BGP Instance, page 1-11
- Restarting a BGP Instance, page 1-13
- Shutting Down BGP, page 1-13
- Configuring BGP Peers, page 1-13
- Configuring Dynamic AS Numbers for Prefix Peers, page 1-15
- Clearing BGP Information, page 1-17

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 the BGP Feature

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

SUMMARY STEPS

1. **configure terminal**
2. **feature bgp**

3. (Optional) **show feature**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`feature bgp`**
	Example:
	`switch(config)# feature bgp`
		Enables the BGP feature.

Step 3	**`show feature`**
	Example:
	`switch(config)# show feature`
		(Optional) Displays enabled and disabled features.

Step 4	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config`
	`startup-config`
		(Optional) Saves this configuration change.

Use the **no feature bgp** command to disable the BGP feature and remove all associated configuration.

Command	Purpose

`no feature bgp`
Example:
`switch(config)# no feature bgp`
	Disables the BGP feature and removes all associated configuration.

Creating a BGP Instance

You can create a BGP instance and assign a router ID to the BGP instance. See the "BGP Router Identifier" section on page 1-3. Cisco NX-OS supports 2-byte or 4-byte autonomous system (AS) numbers in plain-text notation or as.dot notation. See the "4-Byte AS Number Support" section on page 1-2 for more information.

BEFORE YOU BEGIN

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

BGP must be able to obtain a router ID (for example, a configured loopback address).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*

3. (Optional) **router-id** *ip-address*
4. **address-family ipv4** {**unicast** | **multicast**}
5. (Optional) **network** *ip-prefix* [**route-map** *map-name*]
6. (Optional) **show bgp all**
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router bgp** *autonomous-system-number* **Example:** switch(config)# router bgp 64496 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 numbe in xx.xx format.
Step 3	**router-id** *ip-address* **Example:** switch(config-router)# router-id 192.0.2.255	(Optional) Configures the BGP router ID. This IP address identifies this BGP speaker. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 4	**address-family ipv4**{**unicast** | **multicast**} Example: switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Enters global address family configuration mode for the IPv4 or IPv6 address family. This command triggers an automatic notification and session reset for all BGP neighbors.
Step 5	**network** *ip-prefix* [**route-map** *map-name*] **Example**: switch(config-router-af)# network 192.0.2.0	(Optional) Specifies a network as local to this autonomous system and adds it to the BGP routing table. For exterior protocols, the network command controls which networks are advertised. Interior protocols use the network command to determine where to send updates.
Step 6	**show bgp all** **Example**: switch(config-router-af)# show bgp all	(Optional) Displays information about all BGP address families.
Step 7	**copy running-config startup-config** **Example:** switch(config-router-af)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **no router bgp** command to remove the BGP process and the associated configuration.

Command	Purpose
no router bgp *autonomous-system-number* **Example:** switch(config)# no router bgp 201	Deletes the BGP process and the associated configuration.

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:

Command	Purpose
restart bgp instance-tag **Example:** 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 and retain the configuration.

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

Command	Purpose
shutdown **Example:** switch(config-router)# shutdown	Gracefully shuts down BGP.

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

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*
3. **neighbor** *ip-address* **remote-as** *as-number*
4. (Optional) **description** *text*
5. (Optional) **timers** *keepalive-time hold-time*
6. (Optional) **shutdown**
7. **address-family ipv4** {**unicast** | **multicast**}
8. (Optional) **show bgp ipv4** {**unicast** | **multicast**} **neighbors**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router bgp`** *`autonomous-system-number`*
	
	`Example:`
	`switch(config)# router bgp 64496`
	`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 numbe in xx.xx format.

Step 3	**`neighbor`** *`ip-address`* **`remote-as`** *`as-number`*
	
	`Example:`
	`switch(config-router)# neighbor 209.165.201.1 remote-as 64497`
	`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.

Step 4	**`description`** *`text`*
	
	`Example:`
	`switch(config-router-neighbor)# description Peer Router B`
	`switch(config-router-neighbor)#`
		(Optional) Adds a description for the neighbor. The description is an alphanumeric string up to 80 characters.

Step 5	**`timers`** *`keepalive-time hold-time`*
	
	`Example:`
	`switch(config-router-neighbor)# timers 30 90`
		(Optional) 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.

	Command	Purpose
Step 6	**shutdown** **Example:** switch(config-router-neighbor)# shutdown	(Optional) Administratively shuts down this BGP neighbor. This command triggers an automatic notification and session reset for the BGP neighbor sessions.
Step 7	**address-family** {**ipv4** {**unicast** | **multicast**} **Example:** switch(config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	Enters neighbor address family configuration mode for the unicast IPv4 or IPv6 address family.
Step 8	**show bgp** {**ipv4** {**unicast** | **multicast**} **neighbors** **Example**: switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors	(Optional) Displays information about BGP peers.
Step 9	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af) copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure 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-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

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*

3. **neighbor** *prefix* **remote-as route-map** *map-name*
4. (Optional) **show bgp ipv4** {**unicast** | **multicast**} **neighbors**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router bgp** *autonomous-system-number* **Example:** switch(config)# router bgp 64496 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 numbe in xx.xx format.
Step 3	**neighbor** *prefix* **remote-as route-map** *map-name* **Example:** switch(config-router)# neighbor 192.0.2.0/8 remote-as routemap BGPPeers switch(config-router-neighbor)#	Configures the IPv 4 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. The *map-name* can be any case-sensitive, alphanumeric string up to 63 characters.
Step 4	**show bgp ipv4** {**unicast** | **multicast**} **neighbors** **Example:** switch(config-router-neighbor-af)# show bgp ipv4 unicast neighbors	(Optional) Displays information about BGP peers.
Step 5	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af) copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to 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
```

See Chapter 1, “Configuring Route Policy Manager” for information on route maps.

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: • *neighbor*—IPv4 address of a neighbor. • *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. • *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • *prefix*—IPv4 prefix. All neighbors within that prefix are cleared. • *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 ip {**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 ip {**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 ip {**unicast** | **multicast**} {*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: • *neighbor*—IPv4 address of a neighbor. • *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. • *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • *prefix*—IPv4 prefix. All neighbors within that prefix are cleared. • *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 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: • *neighbor*—IPv4 address of a neighbor. • *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. • *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • *prefix*—IPv4 prefix. All neighbors within that prefix are cleared. • *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 bgp dampening [*ip-neighbor* | *ip-prefix*] [**vrf** *vrf-name*]	Clears route flap dampening in one or more networks. The arguments are as follows: • *ip-neighbor*—IPv4 address of a neighbor. • *ip-prefix*—IPv4. All neighbors within that prefix are cleared. • *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 bgp flap-statistics [*ip-neighbor* | *ip-prefix*] [**vrf** *vrf-name*]	Clears route flap statistics in one or more networks. The arguments are as follows: • *ip-neighbor*—IPv4 address of a neighbor. • *ip-prefix*—IPv4. All neighbors within that prefix are cleared. • *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: • *neighbor*—IPv4 address of a neighbor. • *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. • *name*—Peer template name. The name can be any case-sensitive, alphanumeric string up to 64 characters. • *prefix*—IPv4 prefix. All neighbors within that prefix are cleared. • *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 dampeningin one or more networks. The arguments are as follows: • *ip-neighbor*—IPv4 address of a neighbor. • *ip-prefix*—IPv4. All neighbors within that prefix are cleared. • *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 one or more networks. The arguments are as follows: • *ip-neighbor*—IPv4 address of a neighbor. • *ip-prefix*—IPv4. All neighbors within that prefix are cleared. • *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 information, perform the following tasks:

Command	Purpose
show bgp all [**summary**] [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp convergence [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp ip {**unicast** | **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*] **ip** {**unicast** | **multicast**} [*ip-address*] **community-list** *list-nam*e [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp ip {**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 ip {**unicast** | **multicast**} [*ip-address*] **extcommunity-list** *list-nam*e [**exact-match**] [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP extended community list.

Command	Purpose
show bgp ip {**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 ip {**unicast** | **multicast**} [*ip-address*] **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp ip {**unicast** | **multicast**} [*ip-address*] **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.
show bgp ip {**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 ip {**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 ip {**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.
show bgp ip {**unicast** | **multicast**} [*ip-address*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp ip {**unicast** | **multicast**} [*ip-address*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.
show bgp ip {**unicast** | **multicast**} [i*p-address*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp ip {**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 bgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for more information.

Command	Purpose
show ip mbgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for more information.
show running-configuration bgp	Displays the current running BGP configuration.

Displaying BGP Statistics

To display BGP statistics, use the following commands:

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

Configuration Examples for Basic BGP

This example shows a basic BGP configuration: `feature bgp`

```
router bgp 64496
 neighbor 2001:ODB8:0:1::55 remote-as 64496
  address-family ipv4 unicast
    next-hop-self
```

This example shows a basic BGP configuration: `address-family`

```
router bgp 64496
address-family ipv4 unicast
   network 1.1.10 mask 255.255.255.0
neighbor 10.1.1.1 remote-as 64496
   address-family ipv4 unicast
```

Related Topics

The following topics relate to BGP:

- Chapter 1, “Configuring Route Policy Manager.”

Where to Go Next

See Chapter 1, “Configuring Advanced BGP” for details on the following features:

- Peer templates
- Route redistribution
- Route maps

Additional References

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

- Related Documents, page 1-23
- MIBs, page 1-23

Related Documents

Related Topic	Document Title
BGP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

MIBs

MIBs	MIBs Link
BGP4-MIB CISCO-BGP4-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Configuring Advanced BGP

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

This chapter includes the following sections:

Information About Advanced BGP

BGP is an interdomain routing protocol that provides loop-free routing between organizations or autonomous systems. Cisco NX-OS supports BGP version 4. BGP version 4 includes multiprotocol extensions that allow BGP to carry routing information for IP 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 switches called BGP peers. When connecting to an external organization, the router creates external BGP (eBGP) peering sessions. BGP peers within the same organization exchange routing information through internal BGP (iBGP) peering sessions.

This section includes the following topics:

Peer Templates

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

Cisco NX-OS implements three types of peer templates:

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

Authentication

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

Note The MD5 password must be identical between BGP peers.

Route Policies and Resetting BGP Sessions

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

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

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

Note BGP also uses route maps for route redistribution, route aggregation, route dampening, and other features. See Chapter 1, "Configuring Route Policy Manager," for more information on route maps.

eBGP

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

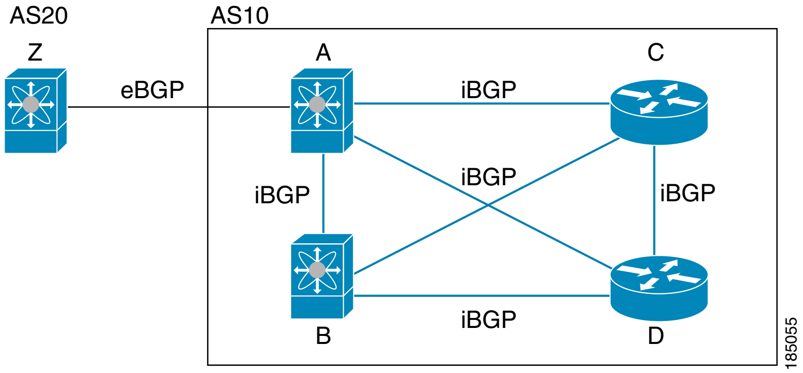
You should use loopback interfaces for establishing eBGP peering sessions because loopback interfaces are less susceptible to interface flapping. An interface *flap* occurs when the interface is administratively brought up or down because of a failure or maintenance issue. See the "Configuring eBGP" section on page 1-22 for information on multihop, fast external failovers, and limiting the size of the AS-path attribute.

iBGP

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

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

Figure 1-1 iBGP Network



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

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

This section includes the following topics:

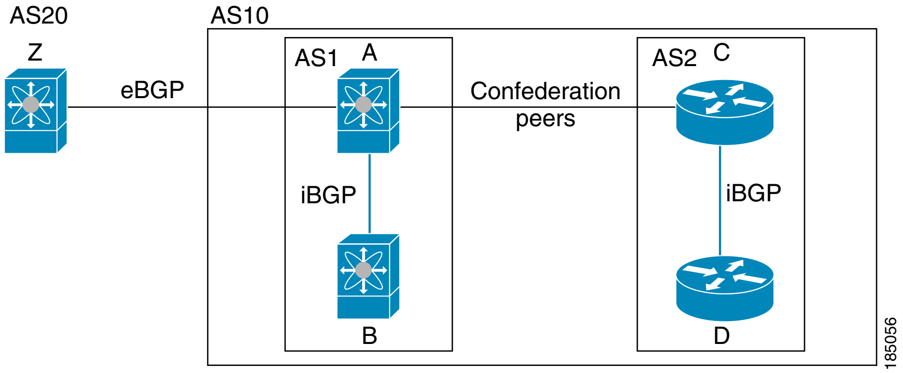
- AS Confederations, page 1-4
- Route Reflector, page 1-5

AS Confederations

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

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

Figure 1-2 AS Confederation



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

Route Reflector

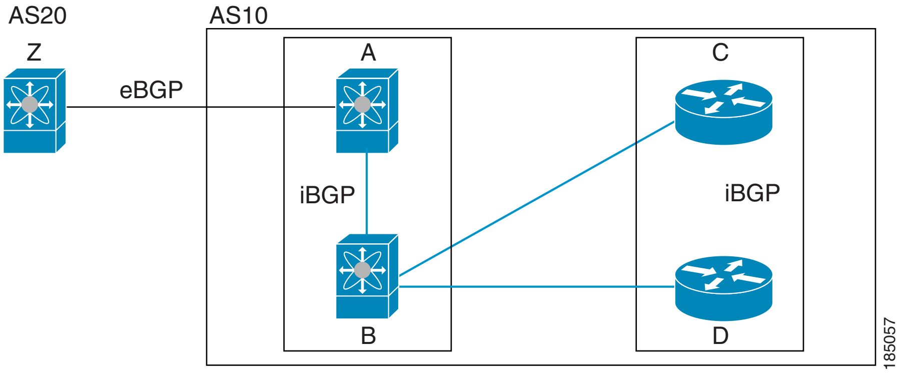
You can alternately reduce the iBGP mesh by using a route reflector configuration. Route reflectors pass learned routes to neighbors so that all iBGP peers do not need to be fully meshed.

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

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

In Figure 1-3, router B is the route reflector. When the route reflector receives routes advertised from router A, it advertises (reflects) the routes to routers C and D. Router A no longer has to advertise to both routers C and D.

Figure 1-3 Route Reflector



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

Capabilities Negotiation

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

If a BGP peer does not support capabilities negotiation, Cisco NX-OS will attempt a new session to the peer without capabilities negotiation if you have configured the address family as IPv4.

Route Dampening

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

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

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

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

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

Load Sharing and Multipath

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

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

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

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

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

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

Route Aggregation

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

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

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

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

BGP Conditional Advertisement

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

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

BGP conditional advertisement adds an exist or not-exist test to each route that matches the configured route map. See the "Configuring BGP Conditional Advertisement" section on page 1-29 for more information.

BGP Next-Hop Address Tracking

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

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

- Next hop becomes unreachable.
- Next hop becomes reachable.
- Fully recursed IGP metric to the next hop changes.
- First hop IP address or first hop interface changes.
- Next hop becomes connected.
- Next hop becomes unconnected.
- Next hop becomes a local address.

- Next hop becomes a nonlocal address.

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

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

- Critical events are related to the reachability (reachable and unreachable), connectivity (connected and unconnected), and locality (local and nonlocal) of the next hops. Notifications for these events are not delayed.
- Noncritical events include only the IGP metric changes.

See the "Configuring BGP Next-Hop Address Tracking" section on page 1-21 for more information.

Route Redistribution

You can configure BGP to redistribute static routes or routes from other protocols. You configure a route policy with the **redistribution to control which routes are passed into BGP**. A route policy allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. See Chapter 1, "Configuring Route Policy Manager," for more information.Prior to Cisco NX-OS Release 5.2(1), when you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an additional deny statement into the route map. iBGP is not redistributed to IGP by default.

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

You can change the default behavoir 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
```

BFD

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

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

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

See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x* for more information.

Tuning BGP

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

This section includes the following topics:

- BGP Timers, page 1-9
- Tuning the Best-Path Algorithm, page 1-9

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 MED attribute and the router ID.

Multiprotocol BGP

BGP on Cisco NX-OS supports multiple address families. Multiprotocol BGP (MP-BGP) carries different sets of routes depending on the address family. For example, BGP can carry one set of routes for IPv4 unicast routing, and one set of routes for IPv4 multicast routing. You can use MP-BGP for reverse-path forwarding (RPF) checks in IP multicast networks.

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

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

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

Virtualization Support

Cisco NX-OS supports multiple instances of BGP that run on the same system.

Licensing Requirements for Advanced BGP

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	BGP requires an LAN Enterprise Services license. For a complete explanation of the Cisco NX-OS licensing scheme and how to obtain and apply licenses, see the *Cisco NX-OS Licensing Guide*. **Note** Make sure the LAN Base Services license is installed on the switch to enable Layer 3 interfaces.

Prerequisites for BGP

BGP has the following prerequisites:

- You must enable the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).
- 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 BGP

BGP has the following configuration guidelines and limitations:

- The dynamic AS number prefix peer configuration overrides the individual AS number configuration inherited from a BGP template.
- If you configure a dynamic AS number for prefix peers in an AS confederation, BGP establishes sessions with only the AS numbers in the local confederation.
- BGP sessions created through a dynamic AS number prefix peer ignore any configured eBGP multihop time-to-live (TTL) value or a disabled check for directly connected peers.
- Configure a router ID for BGP to avoid automatic router ID changes and session flaps.
- Use the maximum-prefix configuration option per peer to restrict the number of routes received and system resources used.
- Configure the update-source to establish a session with eBGP multihop sessions.
- Specify a BGP route map if you configure redistribution.
- Configure the BGP router ID within a VRF.
- Cisco NX-OS does not support multi-hop BFD. BFD for BGP has the following limitations:
  - BFD is supported only for BGP IPv4.
  - BFD is supported only for eBGP peers and iBGP single-hop peers.
  - To enable BFD for iBGP single-hop peers, you must configure the update-source option on the physical interface.
  - BFD is not supported for multi-hop iBGP peers and multi-hop eBGP peers.
- If you decrease the keepalive and hold timer values, the network might experience session flaps.

Default Settings

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

Table 1-1 Default BGP Parameters

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

Configuring Advanced BGP

This section describes how to configure advanced BGP and includes the following topics:

- Configuring BGP Session Templates, page 1-12
- Configuring BGP Peer-Policy Templates, page 1-14
- Configuring BGP Peer Templates, page 1-16
- Configuring Prefix Peering, page 1-19
- Configuring BGP Authentication, page 1-20
- Resetting a BGP Session, page 1-20
- Modifying the Next-Hop Address, page 1-21
- Configuring BGP Next-Hop Address Tracking, page 1-21
- Configuring Next-Hop Filtering, page 1-22
- Disabling Capabilities Negotiation, page 1-22
- Configuring eBGP, page 1-22
- Configuring AS Confederations, page 1-24
- Configuring Route Reflector, page 1-25
- Configuring Route Dampening, page 1-27
- Configuring Load Sharing and ECMP, page 1-27
- Configuring Maximum Prefixes, page 1-28
- Configuring Dynamic Capability, page 1-28
- Configuring Aggregate Addresses, page 1-28
- Configuring BGP Conditional Advertisement, page 1-29
- Configuring Route Redistribution, page 1-31
- Tuning BGP, page 1-32
- Configuring Virtualization, page 1-35

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

Configuring BGP Session Templates

You can use BGP session templates to simplify 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

Note Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the **default** form of the command to reset that attribute to the default state.

SUMMARY STEPS

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

DETAILED STEPS

	Command	Purpose

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

Step 2	**router bgp** *autonomous-system-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enables BGP and assigns the autonomous system number to the local BGP speaker.

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

Step 4	**password** *number password* **Example**: `switch(config-router-stmp)# password 0 test`	(Optional) Adds the clear text password *test* to the neighbor. The password is stored and displayed in type 3 encrypted form (3DES).
Step 5	**timers** *keepalive hold* **Example**: `switch(config-router-stmp)# timers 30 90`	(Optional) Adds the BGP keepalive and holdtimer values to the peer-session template. The default keepalive interval is 60. The default hold time is 180.

Step 6	**exit**
	Example:
	`switch(config-router-stmp)# exit`
	`switch(config-router)#`
		Exits peer-session template configuration mode.

Step 7	**neighbor** *ip-address* **remote-as** *as-number*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65536`
	`switch(config-router-neighbor)#`
		Places the router in the neighbor configuration mode for BGP routing and configures the neighbor IP address.

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

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

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

Use the **show bgp neighbor** command to see the template applied. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for details on all commands available in the template.

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

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer-session BaseSession
switch(config-router-stmp)# timers 30 90
switch(config-router-stmp)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65536
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# description Peer Router A
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor)# copy running-config startup-config
```

Configuring BGP Peer-Policy Templates

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

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

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

BEFORE YOU BEGIN

Note Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the default form of the command to reset that attribute to the default state.

SUMMARY STEPS

1. **configure terminal**
2. **router bgp** *autonomous-system-number*

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

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router bgp`** *`autonomous-system-number`*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enables BGP and assigns the autonomous system number to the local BGP speaker.

Step 3	**`template peer-policy`** *`template-name`*
	Example:
	`switch(config-router)# template`
	`peer-policy BasePolicy`
	`switch(config-router-ptmp)#`
		Creates a peer-policy template.

Step 4	**`advertise-active-only`**
	Example:
	`switch(config-router-ptmp)#`
	`advertise-active-only`
		(Optional) Advertises only active routes to the peer.

Step 5	**`maximum-prefix`** *`number`*
	Example:
	`switch(config-router-ptmp)#`
	`maximum-prefix 20`
		(Optional) Sets the maximum number of prefixes allowed from this peer.

Step 6	**`exit`**
	Example:
	`switch(config-router-ptmp)# exit`
	`switch(config-router)#`
		Exits peer-policy template configuration mode.

Step 7	**`neighbor`** *`ip-address`* **`remote-as`** *`as-number`*
	Example:
	`switch(config-router)# neighbor`
	`192.168.1.2 remote-as 65536`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

	Command	Purpose
Step 8	**address-family ipv4** {**multicast** | **unicast**} **Example**: switch(config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	Enters global address family configuration mode for the IPv4 address family.
Step 9	**inherit peer-policy** *template-name preference* **Example**: switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1	Applies a peer-policy template to the peer address family configuration and assigns the preference value for this peer policy.
Step 10	**show bgp peer-policy** *template-name* **Example:** switch(config-router-neighbor-af)# show bgp peer-policy BasePolicy	(Optional) Displays the peer-policy template.
Step 11	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for details on all commands available in the template.

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

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer-session BasePolicy
switch(config-router-ptmp)# maximum-prefix 20
switch(config-router-ptmp)# exit
switch(config-router)# neighbor 192.168.1.1 remote-as 65536
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy
switch(config-router-neighbor-af)# copy running-config startup-config
```

Configuring BGP Peer Templates

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

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

BEFORE YOU BEGIN

Note Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).When editing a template, you can use the **no** form of a command at either the peer or template level to explicitly override a setting in a template. You must use the default form of the command to reset that attribute to the default state.

SUMMARY STEPS

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

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router bgp`** *`autonomous-system-number`*
	Example:
	`switch(config)# router bgp 65536`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

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

	Command	Purpose
Step 4	**inherit peer-sessio**n *template-name* **Example**: switch(config-router-neighbor)# inherit peer-session BaseSession	(Optional) Inherits a peer-session template in the peer template.
Step 5	**address-family ipv4**{**multicast** | **unicast**} **Example**: switch(config-router-neighbor)# address-family ipv4 unicast switch(config-router-neighbor-af)#	(Optional) Configures the global address family configuration mode for the IPv4 or IPv6 address family.
Step 6	**inherit peer** *template-name* **Example**: switch(config-router-neighbor-af)# inherit peer BasePolicy	(Optional) Applies a peer template to the neighbor address family configuration.
Step 7	**exit** **Example**: switch(config-router-neighbor-af)# exit switch(config-router-neighbor)#	Exits BGP neighbor address family configuration mode.
Step 8	**timers** *keepalive hold* **Example**: switch(config-router-neighbor)# timers 45 100	(Optional) Adds the BGP timer values to the peer. These values override the timer values in the peer-session template, BaseSession.
Step 9	**exit** **Example**: switch(config-router-neighbor)# exit switch(config-router)#	Exits BGP peer template configuration mode.
Step 10	**neighbor** *ip-address* **remote-as** *as-number* **Example:** switch(config-router)# neighbor 192.168.1.2 remote-as 65536 switch(config-router-neighbor)#	Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.
Step 11	**inherit peer** *template-name* **Example**: switch(config-router-neighbor)# inherit peer BasePeer	Inherits the peer template.
Step 12	**timers** *keepalive hold* **Example**: switch(config-router-neighbor)# timers 60 120	(Optional) Adds the BGP timer values to this neighbor. These values override the timer values in the peer template and the peer-session template.
Step 13	**show bgp peer-template** *template-name* **Example:** switch(config-router-neighbor-af)# show bgp peer-template BasePeer	(Optional) Displays the peer template.
Step 14	**copy running-config startup-config** **Example:** switch(config-router-neighbor-af)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **show bgp neighbor** command to see the template applied. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for details on all commands available in the template.

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

```
switch# configure terminal
switch(config)# router bgp 65536
switch(config-router)# template peer BasePeer
switch(config-router-neighbor)# inherit peer-session BaseSession
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# inherit peer-policy BasePolicy 1
switch(config-router-neighbor-af)# exit
switch(config-router-neighbor)# exit
switch(config-router)# neighbor 192.168.1.2 remote-as 65536
switch(config-router-neighbor)# inherit peer BasePeer
switch(config-router-neighbor)# copy running-config startup-config
```

Configuring Prefix Peering

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

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

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

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

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

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

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

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

```
switch(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 ip bgp neighbor** command to show the details of the configuration for that prefix peering with a list of the currently accepted instances and the counts of active, maximum concurrent, and total accepted peers.

Configuring BGP Authentication

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

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

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

Resetting a BGP Session

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

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

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

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

Command	Purpose

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

Modifying the Next-Hop Address

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

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

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

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

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. You can configure the critical timer for routes that affect BGP next-hop reachability, and you can configure the noncritical timer for all other routes in the BGP table.

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

Command	Purpose
nexthop trigger-delay {**critical** | **non-critical**} *milliseconds* **Example**: `switch(config-router-af)# nexthop trigger-delay critical 5000`	Specifies the next-hop address tracking delay timer for critical next-hop reachability routes and for noncritical routes. The range is from 1 to 4294967295 milliseconds. The critical timer default is 3000. The noncritical timer default is 10000.
nexthop route-map *name* **Example**: `switch(config-router-af)# nexthop route-map nextHopLimits`	Specifies a route map to match the BGP next-hop addresses to. The name can be any case-sensitive, alphanumeric string up to 63 characters.

Configuring Next-Hop Filtering

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

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

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

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

Disabling Capabilities Negotiation

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

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

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

Configuring eBGP

This section includes the following topics:

- Disabling eBGP Single-Hop Checking, page 1-22
- Configuring eBGP Multihop, page 1-23
- Disabling a Fast External Failover, page 1-23
- Limiting the AS-path Attribute, page 1-24

Disabling eBGP Single-Hop Checking

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

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

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

Configuring eBGP Multihop

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

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

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

Disabling a Fast External Failover

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

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

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

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 sub-autonomous systems.

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

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

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

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

Limiting the AS-path Attribute

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

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

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

Configuring AS Confederations

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

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

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

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

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

Configuring Route Reflector

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

BEFORE YOU BEGIN

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

SUMMARY STEPS

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

DETAILED STEPS

	Command or Action	Purpose

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

Step 2	**router bgp** *as-number*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

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

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

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

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

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

Step 8	**address-family ipv4** {**unicast** | **multicast**}
	Example:
	`switch(config-router-neighbor)# address-family ipv4 unicast`
	`switch(config-router-neighbor-af)#`
		Enters neighbor address family configuration mode for the unicast IPv4 or IPv6 address family.

Step 9	**route-reflector-client** **Example**: `switch(config-router-neighbor-af)# route-reflector-client`	Configures the switch as a BGP route reflector and configures the neighbor as its client. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

	Command or Action	Purpose
Step 10	**show bgp ip** {**unicast** | **multicast**} **neighbors** **Example:** `switch(config-router-neighbor-af)# show bgp ip unicast neighbors`	(Optional) Displays the BGP peers.
Step 11	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Saves this configuration change.

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

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

Configuring Route Dampening

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

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

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

Configuring Load Sharing and ECMP

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

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

Command	Purpose
maximum-paths [**ibgp**] *maxpaths* **Example:** `switch(config-router-af)# maximum-paths 12`	Configures the maximum number of equal-cost paths for load sharing. The range is from 1 to 64. The default is 8.

Configuring Maximum Prefixes

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

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

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

Configuring Dynamic Capability

You can configure dynamic capability for a BGP peer.

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

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

Configuring Aggregate Addresses

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

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

Command	Purpose

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

Configuring BGP Conditional Advertisement

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

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

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

BEFORE YOU BEGIN

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

SUMMARY STEPS

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

7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`router bgp`** *`as-number`*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Enters BGP mode and assigns the autonomous system number to the local BGP speaker.

Step 3	**`neighbor`** *`ip-address`* **`remote-as`** *`as-number`*
	Example:
	`switch(config-router)# neighbor 192.168.1.2 remote-as 65537`
	`switch(config-router-neighbor)#`
		Places the router in neighbor configuration mode for BGP routing and configures the neighbor IP address.

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

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

	Command	Purpose
Step 6	**show ip bgp neighbor** **Example:** `switch(config-router-neighbor-af)# show ip bgp neighbor`	(Optional) Displays information about BGP and the configured conditional advertisement route maps.
Step 7	**copy running-config startup-config** **Example:** `switch(config-router-neighbor-af)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure BGP conditional advertisement:

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

Note Redistribution does not work if the access list is used as a **match** option in **route-maps.**

BEFORE YOU BEGIN

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

SUMMARY STEPS

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

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router bgp** *as-number* Example: switch(config)# router bgp 65536 switch(config-router)#	Enters BGP mode and assigns the autonomous system number to the local BGP speaker.
Step 3	**address-family ipv4** {**unicast** | **multicast**} **Example**: switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Enters address family configuration mode.
Step 4	**redistribute** {**direct** | {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example**: switch(config-router-af)# redistribute eigrp 201 route-map Eigrpmap	Redistributes routes from other protocols into BGP. See the "Configuring Route Maps" section on page 1-12 for more information about route maps.
Step 5	**default-metric** *value* **Example**: switch(config-router-af)# default-metric 33	(Optional) Generates a default route into BGP.
Step 6	**copy running-config startup-config** **Example:** switch(config-router-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to redistribute EIGRP into BGP:

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

Tuning BGP

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

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

Command	Purpose
bestpath [**always-compare-med** | **compare-routerid** | **med** {**missing-as-worst** | **non-deterministic**}] **Example**: switch(config-router)# bestpath always-compare-med	Modifies the best-path algorithm. The optional parameters are as follows: • **always-compare-med**—Compares MED on paths from different autonomous systems. • **compare-routerid**—Compares the router IDs for identical eBGP paths. • **med missing-as-worst**— Sees a missing MED as the highest MED. • **med non-deterministic**—Does not always select the best MED path from among the paths from the same autonomous system.
enforce-first-as **Example:** switch(config-router)# enforce-first-as	Enforces the neighbor autonomous system to be the first AS number listed in the AS_path attribute for eBGP.
log-neighbor-changes **Example**: switch(config-router)# log-neighbor-changes	Generates a system message when a neighbor changes state.
router-id *id* **Example**: switch(config-router)# router-id 209.165.20.1	Manually configures the router ID for this BGP speaker.
timers [**bestpath-delay** *delay* | **bgp** *keepalive holdtime* | **prefix-peer-timeout** *timeout*] **Example**: switch(config-router)# timers bgp 90 270	Sets the BGP timer values. The optional parameters are as follows: • *delay*—Initial best-path timeout value after a restart. The range is from 0 to 3600 seconds. The default value is 300. • keepalive—BGP session keepalive time. The range is from 0 to 3600 seconds. The default value is 60. • holdtime—BGP session hold time.The range is from 0 to 3600 seconds. The default value is 180. • timeout—Prefix peer timeout value. The range is from 0 to 1200 seconds. The default value is 30. You must manually reset the BGP sessions after configuring this command.

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

Command	Purpose
distance *ebgp-distance ibgp distance local-distance* **Example:** `switch(config-router-af)# distance 20 100 200`	Sets the administrative distance for BGP. The range is from 1 to 255. The defaults are as follows: • eBGP distance—20. • iBGP distance—200. • local distance—220. Local distance is the administrative distance used for aggregate discard routes when they are installed in the RIB.

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

Command	Purpose
description *string* **Example:** `switch(config-router-neighbor)# description main site`	Sets a descriptive string for this BGP peer. The string can be up to 80 alphanumeric characters.
low-memory exempt **Example:** `switch(config-router-neighbor)# low-memory exempt`	Exempts this BGP neighbor from a possible shutdown due to a low memory condition.
transport connection-mode passive **Example:** `switch(config-router-neighbor)# transport connection-mode passive`	Allows a passive connection setup only. This BGP speaker does not initiate a TCP connection to a BGP peer. You must manually reset the BGP sessions after configuring this command.
remove-private-as **Example:** `switch(config-router-neighbor)# remove-private-as`	Removes private AS numbers from outbound route updates to an eBGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
update-source *interface-type number* **Example:** `switch(config-router-neighbor)# update-source ethernet 2/1`	Configures the BGP speaker to use the source IP address of the configured interface for BGP sessions to the peer. This command triggers an automatic notification and session reset for the BGP neighbor sessions.

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

Command	Purpose
suppress-inactive **Example**: `switch(config-router-neighbor-af)# suppress-inactive`	Advertises the best (active) routes only to the BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
default-originate [**route-map** *map-name*] **Example**: `switch(config-router-neighbor-af)# default-originate`	Generates a default route to the BGP peer.

filter-list *list-name* {**in** | **out**}
Example:
`switch(config-router-neighbor-af)# filter-list BGPFilter in`
	Applies an AS_path filter list to this BGP peer for inbound or outbound route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

prefix-list *list-name* {**in** | **out**}
Example:
`switch(config-router-neighbor-af)# prefix-list PrefixFilter in`
	Applies a prefix list to this BGP peer for inbound or outbound route updates. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

send-community **Example**: `switch(config-router-neighbor-af)# send-community`	Sends the community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.
send-extcommunity **Example**: `switch(config-router-neighbor-af)# send-extcommunity`	Sends the extended community attribute to this BGP peer. This command triggers an automatic soft clear or refresh of BGP neighbor sessions.

Configuring Virtualization

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

BEFORE YOU BEGIN

Ensure that you have enabled the BGP feature (see the "Enabling the BGP Feature" section on page 1-10).

SUMMARY STEPS

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

DETAILED STEPS

	Command	Purpose

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

Step 2	**`vrf context`** *`vrf-name`*
	Example:
	`switch(config)# vrf context RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

Step 3	**`exit`**
	Example:
	`switch(config-vrf)# exit`
	`switch(config)#`
		Exits VRF configuration mode.

Step 4	**`router bgp`** *`as-number`*
	Example:
	`switch(config)# router bgp 65536`
	`switch(config-router)#`
		Creates a new BGP process with the configured autonomous system number.

Step 5	**`vrf`** *`vrf-name`*
	Example:
	`switch(config-router)# vrf RemoteOfficeVRF`
	`switch(config-router-vrf)#`
		Enters the router VRF configuration mode and associates this BGP instance with a VRF.

Step 6	**`neighbor`** *`ip-address`* **`remote-as`** *`as-number`*
	Example:
	`switch(config-router-vrf)# neighbor 209.165.201.1 remote-as 65536`
	`switch(config-router--vrf-neighbor)#`
		Configures the IP address and AS number for a remote BGP peer.

Step 7	**`copy running-config startup-config`**
	Example:
	`switch(config-router-vrf-neighbor)# copy running-config startup-config`
		(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# vrf context NewVRF
switch(config-vrf)# exit
switch(config)# router bgp 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 information, perform one of the following tasks:

Command	Purpose
show bgp all [**summary**] [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp convergence [**vrf** *vrf-name*]	Displays the BGP information for all address families.
show bgp ip {**unicast** | **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*] **ip** {**unicast** | **multicast**} [*ip-address*] **community-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match a BGP community list.
show bgp ip {**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 ip {**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 ip {**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 ip {**unicast** | **multicast**} [*ip-address*] **history-paths** [**regexp** *expression*] [**vrf** *vrf-name*]	Displays the BGP route history paths.
show bgp ip {**unicast** | **multicast**} [*ip-address*] **filter-list** *list-name* [**vrf** *vrf-name*]	Displays the information for the BGP filter list.
show bgp ip {**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 ip {**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 ip {**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.
show bgp ip {**unicast** | **multicast**} [*ip-address*] **prefix-list** *list-name* [**vrf** *vrf-name*]	Displays the BGP routes that match the prefix list.
show bgp ip {**unicast** | **multicast**} [*ip-address*] **received-paths** [**vrf** *vrf-name*]	Displays the BGP paths stored for soft reconfiguration.

Command	Purpose
show bgp ip {**unicast** | **multicast**} [*ip-address*] **regexp** *expression* [**vrf** *vrf-name*]	Displays the BGP routes that match the AS_path regular expression.
show bgp ip {**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 bgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for more information.
show ip mbgp *options*	Displays the BGP status and configuration information. This command has multiple options. See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for more information.
show running-configuration bgp	Displays the current running BGP configuration.

Displaying BGP Statistics

To display BGP statistics, use the following commands:

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

Related Topics

The following topics can give more information on BGP:

- Chapter 1, “Configuring Advanced BGP”
- Chapter 1, “Configuring Route Policy Manager”

Additional References

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

- Related Documents, page 1-39
- MIBs, page 1-39

Related Documents

Related Topic	Document Title
BGP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

MIBs

MIBs	MIBs Link
BGP4-MIB CISCO-BGP4-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Configuring RIP

This chapter describes how to configure the Routing Information Protocol (RIP).

This chapter includes the following sections:

Information About RIP

This section includes the following topics:

RIP Overview

RIP uses User Datagram Protocol (UDP) data packets to exchange routing information in small internetworks. RIPv2 supports IPv4. RIPv2 uses an optional authentication feature supported by the RIPv2 protocol (see the "RIPv2 Authentication" section on page 1-2).

RIP uses the following two message types:

- Request—Sent to the multicast address 224.0.0.9 to request route updates from other RIP-enabled routers.
- Response—Sent every 30 seconds by default (see the "Verifying the RIP Configuration" section on page 1-17). The router also sends response messages after it receives a Request message. The response message contains the entire RIP route table. RIP sends multiple response packets for a request if the RIP routing table cannot fit in one response packet.

RIP uses a *hop count* for the routing metric. The hop count is the number of routers that a packet can traverse before reaching its destination. A directly connected network has a metric of 1; an unreachable network has a metric of 16. This small range of metrics makes RIP an unsuitable routing protocol for large networks.

RIPv2 Authentication

You can configure authentication on RIP messages to prevent unauthorized or invalid routing updates in your network. Cisco NX-OS supports a simple password or an MD5 authentication digest.

You can configure the RIP authentication per interface by using key-chain management for the authentication keys. Key-chain management allows you to control changes to the authentication keys used by an MD5 authentication digest or simple text password authentication. See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0,* for more details about creating key-chains.

To use an MD5 authentication digest, you configure a password that is shared at the local router and all remote RIP neighbors. Cisco NX-OS creates an MD5 one-way message digest based on the message itself and the encrypted password and sends this digest with the RIP message (Request or Response). The receiving RIP neighbor validates the digest by using the same encrypted password. If the message has not changed, the calculation is identical and the RIP message is considered valid.

An MD5 authentication digest also includes a sequence number with each RIP message to ensure that no message is replayed in the network.

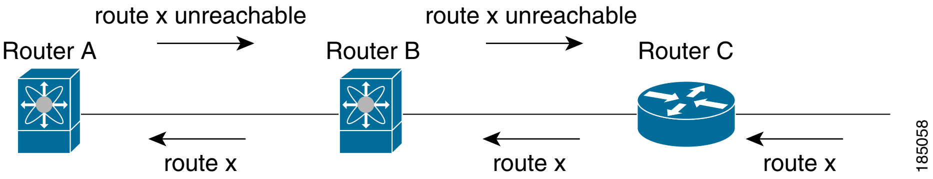
Split Horizon

You can use split horizon to ensure that RIP never advertises a route out of the interface where it was learned.

Split horizon is a method that controls the sending of RIP update and query packets. When you enable split horizon on an interface, Cisco NX-OS does not send update packets for destinations that were learned from this interface. Controlling update packets in this manner reduces the possibility of routing loops.

You can use split horizon with poison revers to configure an interface to advertise routes learned by RIP as unreachable over the interface that learned the routes. Figure 1-1 shows a sample RIP network with split horizon with poison reverse enabled.

Figure 1-1 RIP with Split Horizon Poison Reverse



Router C learns about route X and advertises that route to router B. Router B in turn advertises route X to router A, but sends a route X unreachable update back to router C.

By default, split horizon is enabled on all interfaces.

Route Filtering

You can configure a route policy on a RIP-enabled interface to filter the RIP updates. Cisco NX-OS updates the route table with only those routes that the route policy allows.

Route Summarization

You can configure multiple summary aggregate addresses for a specified interface. Route summarization 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 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.

If more specific routes are in the routing table, RIP advertises the summary address from the interface with a metric equal to the maximum metric of the more specific routes.

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

Route Redistribution

You can use RIP to redistribute static routes or routes from other protocols. When you configure redistribution, use a route policy to control which routes are passed into RIP. A route policy allows you to filter routes based on attributes such as the destination, origination protocol, route type, route tag, and so on. For more information, see Chapter 1, "Configuring Route Policy Manager."

Whenever you redistribute routes into a RIP routing domain, by default Cisco NX-OS does not redistribute the default route into the RIP routing domain. You can generate a *default route* into RIP, which can be controlled by a route policy.

You also configure the default metric that is used for all imported routes into RIP.

Load Balancing

You can use load balancing to allow a router to distribute traffic over all the router network ports that are the same distance from the destination address. Load balancing increases the utilization of network segments and increases effective network bandwidth.

Cisco NX-OS supports the Equal Cost Multiple Paths (ECMP) feature with up to 16 equal-cost paths in the RIP route table and the unicast RIB. You can configure RIP to load balance traffic across some or all of those paths.

Virtualization Support

Cisco NX-OS supports multiple instances of the RIP protocol that runs on the same system. RIP supports Virtual Routing and Forwarding instances (VRFs).

By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. See Chapter 1, "Configuring Layer 3 Virtualization."

Licensing Requirements for RIP

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	RIP requires a LAN Base Services license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*. **Note** Make sure the LAN Base Services license is installed on the switch to enable Layer 3 interfaces.

Prerequisites for RIP

RIP has the following prerequisites:

- You must enable the RIP feature (see the "Enabling the RIP Feature" section on page 1-5).

Guidelines and Limitations

RIP has the following configuration guidelines and limitations:

- Cisco NX-OS does not support RIPv1. If Cisco NX-OS receives a RIPv1 packet, it logs a message and drops the packet.
- Cisco NX-OS does not establish adjacencies with RIPv1 routers.

Default Settings

Table 1-1 lists the default settings for RIP parameters.

Table 1-1 Default RIP Parameters

Parameters	Default
Maximum paths for load balancing	16
Split horizon	Enabled

Configuring RIP

This section includes the following topics:

- Enabling the RIP Feature, page 1-5
- Creating a RIP Instance, page 1-6
- Configuring RIP on an Interface, page 1-8
- Configuring a Passive Interface, page 1-11
- Configuring Route Summarization, page 1-11
- Configuring Route Summarization, page 1-11
- Configuring Route Redistribution, page 1-12
- Configuring Virtualization, page 1-13
- Tuning RIP, page 1-16

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 the RIP Feature

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

SUMMARY STEPS

1. **configure terminal**
2. **feature rip**
3. (Optional) **show feature**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`feature rip`**
	Example:
	`switch(config)# feature rip`
		Enables the RIP feature.

Step 3	**`show feature`**
	Example:
	`switch(config)# show feature`
		(Optional) Displays enabled and disabled features.

Step 4	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

Use the **no feature rip** command to disable the RIP feature and remove all associated configuration.

Command	Purpose

`no feature rip`
Example:
`switch(config)# no feature rip`
	Disables the RIP feature and removes all associated configuration.

Creating a RIP Instance

You can create a RIP instance and configure the address family for that instance.

BEFORE YOU BEGIN

Ensure that you have enabled the RIP feature (see the "Enabling the RIP Feature" section on page 1-5).

SUMMARY STEPS

1. **configure terminal**
2. **router rip** *instance-tag*
3. **address-family ipv4 unicast**
4. (Optional) **show ip rip** [**instance** *instance-tag*] [**vrf** *vrf-name*]
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router rip** *instance-tag* **Example:** switch(config)# router RIP Enterprise switch(config-router)#	Creates a new RIP instance with the configured *instance-tag*.
Step 3	**address-family ipv4 unicast** **Example:** switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Configures the address family for this RIP instance and enters address-family configuration mode.
Step 4	**show ip rip** [**instance** *instance-tag*] [**vrf** *vrf-name*] **Example**: switch(config-router-af)# show ip rip	(Optional) Displays a summary of RIP information for all RIP instances.
Step 5	**copy running-config startup-config** **Example:** switch(config-router-af)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **no router rip** command to remove the RIP instance and the associated configuration.

Command	Purpose
no router rip *instance-tag* **Example:** switch(config)# no router rip Enterprise	Deletes the RIP instance and all associated configuration.

Note You must also remove any RIP commands configured in interface mode.

You can configure the following optional parameters for RIP in address-family configuration mode:

Command	Purpose
distance *value* **Example:** switch(config-router-af)# distance 30	Sets the administrative distance for RIP. The range is from 1 to 255. The default is 120. See the "Administrative Distance" section on page 1-7.
maximum-paths *number* **Example:** switch(config-router-af)# maximum-paths 6	Configures the maximum number of equal-cost paths that RIP maintains in the route table. The range is from 1 to 16. The default is 16.

This example shows how to create a RIP instance for IPv4 and set the number of equal-cost paths for load balancing:

```
switch# configure terminal
switch(config)# router rip Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# max-paths 10
switch(config-router-af)# copy running-config startup-config
```

Restarting a RIP Instance

You can restart a RIP instance. This clears all neighbors for the instance.

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

Command	Purpose
restart rip *instance-tag* **Example:** `switch(config)# restart rip Enterprise`	Restarts the RIP instance and removes all neighbors.

Configuring RIP on an Interface

You can add an interface to a RIP instance.

BEFORE YOU BEGIN

Ensure that you have enabled the RIP feature (see the "Enabling the RIP Feature" section on page 1-5).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **ip router rip** *instance-tag*
5. (Optional) **show ip rip** [**instance** *instance-tag*] **interface** [*interface-type slot/port*] [**vrf** *vrf-name*] [**detail**]
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**interface** *interface-type slot/port* **Example:** switch(config)# interface ethernet 1/2 switch(config-if)#	Enters interface configuration mode. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 3	**no switchport** **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 4	**ip router rip** *instance-tag* **Example**: switch(config-if)# ip router rip Enterprise	Associates this interface with a RIP instance.
Step 5	**show ip rip** [**instance** *instance-tag*] **interface** [*interface-type slot/port*] [**vrf** *vrf-name*] [**detail**] **Example**: switch(config-if)# show ip rip Enterprise tethernet 1/2	(Optional) Displays RIP information for an interface. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 6	**copy running-config startup-config** **Example:** switch(config-if)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to add the Ethernet 1/2 interface to a RIP instance:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip router rip Enterprise
switch(config)# copy running-config startup-config
```

Configuring RIP Authentication

You can configure authentication for RIP packets on an interface.

BEFORE YOU BEGIN

Ensure that you have enabled the RIP feature (see the “Enabling the RIP Feature” section on page 1-5).

Configure a key chain if necessary before enabling authentication. See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0,* for details on implementing key chains.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **ip rip authentication mode**{**text** | **md5**}
5. **ip rip authentication key-chain** *key*
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`interface-type slot/port`*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	**`ip rip authentication mode`** {**`text`** | **`md5`**} **Example**: `switch(config-if)# ip rip authentication mode md5`	Sets the authentication type for RIP on this interface as cleartext or MD5 authentication digest.

Step 5	**`ip rip authentication key-chain`** *`key`*
	Example:
	`switch(config-if)# ip rip authentication keychain RIPKey`
		Configures the authentication key used for RIP on this interface.

Step 6	**`copy running-config startup-config`**
	Example:
	`switch(config-if)# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to create a key chain and configure MD5 authentication on a RIP interface:

```
switch# configure terminal
switch(config)# key chain RIPKey
switch(config)# key-string myrip
switch(config)# accept-lifetime 00:00:00 Jan 01 2000 infinite
switch(config)# send-lifetime 00:00:00 Jan 01 2000 infinite
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip rip authentication mode md5
switch(config-if)# ip rip authentication keychain RIPKey
switch(config-if)# copy running-config startup-config
```

Configuring a Passive Interface

You can configure a RIP interface to receive routes but not send route updates by setting the interface to passive mode.

To configure a RIP interface in passive mode, use the following command in interface configuration mode:

Command	Purpose
ip rip passive-interface **Example**: `switch(config-if)# ip rip passive-interface`	Sets the interface into passive mode.

Configuring Split Horizon with Poison Reverse

You can configure an interface to advertise routes learned by RIP as unreachable over the interface that learned the routes by enabling poison reverse.

To configure split horizon with poison reverse on an interface, use the following command in interface configuration mode:

Command	Purpose
ip rip poison-reverse **Example**: `switch(config-if)# ip rip poison-reverse`	Enables split horizon with poison reverse. Split horizon with poison reverse is disabled by default.

Configuring Route Summarization

You can create aggregate addresses that are represented in the routing table by a summary address. Cisco NX-OS advertises the summary address metric that is the smallest metric of all the more-specific routes.

To configure a summary address on an interface, use the following command in interface configuration mode:

Command	Purpose
ip rip summary-address *ip-prefix/mask-len* **Example:** `switch(config-if)# ip router rip summary-address 192.0.2.0/24`	Configures a summary address for RIP for IPv4 addresses.

Configuring Route Redistribution

You can configure RIP to accept routing information from another routing protocol and redistribute that information through the RIP network. Redistributed routes can optionally be assigned a default route.

Note Redistribution does not work if the access list is used as a **match** option in **route-maps.**

BEFORE YOU BEGIN

Ensure that you have enabled the RIP feature (see the "Enabling the RIP Feature" section on page 1-5).

Configure a route map before configuring redistribution. See the "Configuring Route Maps" section on page 1-12 for details on configuring route maps.

SUMMARY STEPS

1. **configure terminal**
2. **router rip** *instance-tag*
3. **address-family ipv4 unicast**
4. **redistribute** {**bgp** *as* | **direct** | **eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
5. (Optional) **default-information originate** [**always**] [**route-map** *map-name*]
6. (Optional) **default-metric** *value*
7. (Optional) **show ip rip route** [{*ip-prefix* [**longer-prefixes** | **shorter-prefixes**]] [**vrf** *vrf-name*] [**summary**]
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**router rip** *instance-tag*
	Example:
	`switch(config)# router rip Enterprise`
	`switch(config-router)#`
		Creates a new RIP instance with the configured *instance-tag*.

	Command	Purpose
Step 3	**address-family ipv4 unicast** **Example**: switch(config-router)# address-family ipv4 unicast switch(config-router-af)#	Enters address family configuration mode.
Step 4	**redistribute** {**bgp** *as* | **direct** |{**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name* **Example**: switch(config-router-af)# redistribute eigrp 201 route-map RIPmap	Redistributes routes from other protocols into RIP. See the "Configuring Route Maps" section on page 1-12 for more information about route maps.
Step 5	**default-information originate** [**always**] [**route-map** *map-name*] **Example**: switch(config-router-af)# default-information originate always	(Optional) Generates a default route into RIP, optionally controlled by a route map.
Step 6	**default-metric** *value* **Example**: switch(config-router-af)# default-metric 10	(Optional) Sets the default metric for all redistributed routes. The range is from 1 to 15. The default is 1.
Step 7	**show ip rip route** [*ip-prefix* [**longer-prefixes** | **shorter-prefixes**] [**vrf** *vrf-name]* [**summary**] **Example:** switch(config-router-af)# show ip rip route	(Optional) Shows the routes in RIP.
Step 8	**copy running-config startup-config** **Example:** switch(config-router-af)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to redistribute EIGRP into RIP:

```
switch# configure terminal
switch(config)# router rip Enterprise
switch(config-router)# address-family ipv4 unicast
switch(config-router-af)# redistribute eigrp 201 route-map RIPmap
switch(config-router-af)# copy running-config startup-config
```

Configuring Virtualization

You can create multiple VRFs and use the same or multiple RIP instances in each VRF. You assign a RIP 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 RIP feature (see the "Enabling the RIP Feature" section on page 1-5).

SUMMARY STEPS

1. **configure terminal**
2. **vrf** *vrf-name*
3. **exit**
4. **router rip** *instance-tag*
5. **vrf context** *vrf_name*
6. (Optional) **address-family ipv4 unicast**
7. (Optional) **redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
8. **interface ethernet** *slot/port*
9. **no switchport**
10. **vrf member** *vrf-name*
11. **ip-address** *ip-prefix/length*
12. **ip router rip** *instance-tag*
13. (Optional) **show ip rip** [**instance** *instance-tag*] **interface** [*interface-type slot/port*] [**vrf** *vrf-name*]
14. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**vrf** *vrf-name* **Example:** switch(config)# vrf RemoteOfficeVRF switch(config-vrf)#	Creates a new VRF.
Step 3	**exit** **Example:** switch(config-vrf)# exit switch(config)#	Exits VRF configuration mode.
Step 4	**router rip** *instance-tag* **Example:** switch(config)# router rip Enterprise switch(config-router)#	Creates a new RIP instance with the configured instance tag.

	Command	Purpose

Step 5	**vrf context** *vrf-name*
	
	Example:
	`switch(config)# vrf context`
	`RemoteOfficeVRF`
	`switch(config-vrf)#`
		Creates a new VRF and enters VRF configuration mode.

Step 6	**address-family ipv4 unicast**
	
	Example:
	`switch(config-router-vrf)#`
	`address-family ipv4 unicast`
	`switch(config-router-vrf-af)#`
		(Optional) Configures the VRF address family for this RIP instance.

Step 7	**redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} **route-map** *map-name*
	
	Example:
	`switch(config-router-vrf-af)#`
	`redistribute eigrp 201 route-map RIPmap`
		(Optional) Redistributes routes from other protocols into RIP. See the "Configuring Route Maps" section on page 1-12 for more information about route maps.

Step 8	**interface ethernet** *slot/port*
	
	Example:
	`switch(config-router-vrf-af)# interface`
	`ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 9	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.

Step 10	**vrf member** *vrf-name*
	
	Example:
	`switch(config-if)# vrf member`
	`RemoteOfficeVRF`
		Adds this interface to a VRF.

Step 11	**ip address** *ip-prefix/length*
	
	Example:
	`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 12	**ip router rip** *instance-tag*
	
	Example:
	`switch(config-if)# ip router rip`
	`Enterprise`
		Associates this interface with a RIP instance.

Step 13	**show ip rip** [**instance i***nstance-tag*] **interface** [*interface-type slot/port*] [**vrf** *vrf-name*]
	
	Example:
	`switch(config-if)# show ip rip`
	`Enterprise ethernet 1/2`
		(Optional) Displays RIP information for an interface. in a VRF.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 14	**copy running-config startup-config**
	
	Example:
	`switch(config-if)# copy running-config`
	`startup-config`
		(Optional) Saves this configuration change.

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 rip Enterprise
switch(config-router)# vrf RemoteOfficeVRF
switch(config-router-vrf)# address-family ipv4 unicast
switch(config-router-vrf-af)# redistribute eigrp 201 route-map RIPmap
switch(config-router-vrf-af)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# vrf member RemoteOfficeVRF
switch(config-if)# ip address 192.0.2.1/16
switch(config-if)# ip router rip Enterprise
switch(config-if)# copy running-config startup-config
```

Tuning RIP

You can tune RIP to match your network requirements. RIP uses several timers that determine the frequency of routing updates, the length of time before a route becomes invalid, and other parameters. You can adjust these timers to tune routing protocol performance to better suit your internetwork needs.

Note You must configure the same values for the RIP timers on all RIP-enabled routers in your network.

You can use the following optional commands in address-family configuration mode to tune RIP:

Command	Purpose
timers basic *update timeout holddown garbage-collection* **Example:** `switch(config-router-af)# timers basic 40 120 120 100`	Sets the RIP timers in seconds. The parameters are as follows: • update—The range is from 5 to any positive integer. The default is 30. • timeout—The time that Cisco NX-OS waits before declaring a route as invalid. If Cisco NX-OS does not receive route update information for this route before the timeout interval ends, Cisco NX-OS declares the route as invalid. The range is from 1 to any positive integer. The default is 180. • holddown—The time during which Cisco NX-OS ignores better route information for an invalid route. The range is from 0 to any positive integer. The default is 180. • garbage-collection—The time from when Cisco NX-OS marks a route as invalid until Cisco NX-OS removes the route from the routing table. The range is from 1 to any positive integer. The default is 120.

You can use the following optional commands in interface configuration mode to tune RIP:

Command	Purpose
ip rip metric-offset *value* **Example:** `switch(config-if)# ip rip metric-offset 10`	Adds a value to the metric for every router received on this interface. The range is from 1 to 15. The default is 1.

ip rip route-filter {**prefix-list** *list-name* | **route-map** *map-name*| [**in** | **out**]}
Example:
`switch(config-if)# ip rip route-filter route-map InputMap in`
	Specifies a route map to filter incoming or outgoing RIP updates.

Verifying the RIP Configuration

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

Command	Purpose
show ip rip instance [*instance-tag*] [**vrf** *vrf-name*]	Displays the status for an instance of RIP.
show ip rip [**instance** *instance-tag*] **interface** *slot/port* **detail** [**vrf** *vrf-name*]	Displays the RIP status for an interface. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
show ip rip [**instance** *instance-tag*] **neighbor** [*interface-type number*] [**vrf** *vrf-name*]	Displays the RIP neighbor table.
show ip} **rip** [**instance** *instance-tag*] **route** [*ip-prefix/lengh* [**longer-prefixes** | **shorter--prefixes]] [summary**] [**vrf** *vrf-name*]	Displays the RIP route table.
show running-configuration rip	Displays the current running RIP configuration.

Displaying RIP Statistics

To display the RIP statistics, use the following commands:

Command	Purpose
show ip rip [**instance** *instance-tag*] **policy statistics redistribute** {**bgp** *as* | **direct** | {**eigrp** | **ospf** | **ospfv3** | **rip**} *instance-tag* | **static**} [**vrf** *vrf-name*]	Displays the RIP policy status.
show ip rip [**instance** *instance-tag*] **statistics** *interface-type number*] [**vrf** *vrf-name*]	Displays the RIP statistics.

Use the **clear ip rip policy** command to clear policy statistics.

Use the **clear ip rip statistics** command to clear RIP statistics.

Configuration Examples for RIP

This example creates the Enterprise RIP instance in a VRF and adds Ethernet interface 1/2 to this RIP instance. The example also configures authentication for Ethernet interface 1/2 and redistributes EIGRP into this RIP domain.

```
vrf context NewVRF
!
 feature rip
 router rip Enterprise
  vrf NewVRF
   address-family ip unicast
    redistribute eigrp 201 route-map RIPmap
    max-paths 10
!
interface ethernet 1/2
 no switchport
 vrf NewVRF
 ip address 192.0.2.1/16
 ip router rip Enterprise
 ip rip authentication mode md5
 ip rip authentication keychain RIPKey
```

Related Topics

See Chapter 1, “Configuring Route Policy Manager” for more information on route maps.

Additional References

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

- Related Documents, page 1-19
- Standards, page 1-19

Related Documents

Related Topic	Document Title
RIP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

CHAPTER 1

Configuring Static Routing

This chapter describes how to configure static routing on the switch.

This chapter includes the following sections:

Information 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 may 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).

This section includes the following topics:

- Fully Specified Static Routes, page 1-2
- Floating Static Routes, page 1-2
- Remote Next Hops for Static Routes, page 1-3
- BFD, page 1-3
- Virtualization Support, page 1-3

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 need to 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 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, only for point-to-point interfaces. For broadcast interfaces, the next-hop must be an IPv4or IPv6 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 (nondirectly 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 hop(s) that have reachability to the remote next hops.

BFD

Bidirectional forwarding detection (BFD) is supported for static routes. BFD is a detection protocol that provides fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules. See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x* for more information.

Virtualization Support

Static routes support Virtual Routing and Forwarding instances (VRFs).

Licensing Requirements for Static Routing

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Static routing requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*. **Note** Make sure the LAN Base Services license is installed on the switch to enable Layer 3 interfaces.

Prerequisites for Static Routing

Static routing has the following prerequisites:

- The next-hop address for a static route must be reachable or the static route will not be added to the unicast routing table.

Guidelines and Limitations

Static routing has the following configuration guidelines and limitations:

- You can specify an interface as the next-hop address for a static route only for point-to-point interfaces such as GRE tunnels.

Default Settings

Table 1-1 lists the default settings for static routing parameters.

Table 1-1 Default Static Routing Parameters

Parameters	Default
administrative distance	1

Configuring Static Routing

This section includes the following topics:

- Configuring a Static Route, page 1-4
- Configuring Virtualization, page 1-5

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 router.

SUMMARY STEPS

1. **configure terminal**
2. **ip route** {*ip-prefix* | *ip-addr ip-mask*} {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**tag** *tag-value* [*pref*]]
3. (Optional) **show ip static-route**
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`ip route`** `{`*`ip-prefix`*` | `*`ip-addr ip-mask`*`} {[`*`next-hop`*` | `*`nh-prefix`*`] | [`*`interface next-hop`*` | `*`nh-prefix`*`]} [`**`tag`**` `*`tag-value`*` [`*`pref`*`]`
	Example:
	`switch(config)# 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.

	Command	Purpose
Step 3	**show** {**ip static-route** **Example:** `switch(config)# show ip static-route`	(Optional) Displays information about static routes.
Step 4	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure a static route:

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

Use the **no ip static-route** command to remove the static route.

Configuring Virtualization

You can configure a static route in a VRF.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *vrf-name*
3. **ip route** {*ip-prefix* | *ip-addr ip-mask*} {*next-hop* | *nh-prefix* | *interface*} [**tag** *tag-value* [*pref*]]
4. (Optional) **show ip static-route vrf** *vrf-nam*e
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**vrf context** *vrf-name* **Example:** `switch(config)# vrf context StaticVrf`	Creates a VRF and enters VRF configuration mode.
Step 3	**ip route** {*ip-prefix* | *ip-addr ip-mask*} {*next-hop* | *nh-prefix* | *interface*} [**tag** *tag-value* [*pref*] **Example**: `switch(config-vrf)# ip route 192.0.2.0/8 ethernet 1/2`	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.

	Command	Purpose
Step 4	**show ip static-route vrf** *vrf-name* **Example:** `switch(config-vrf)# show ip static-route`	(Optional) Displays information on static routes.
Step 5	**copy running-config startup-config** **Example:** `switch(config-vrf)# copy running-config startup-config`	(Optional) Saves this configuration change.

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
```

Verifying the Static Routing Configuration

To display the static routing configuration information, use this command:

Command	Purpose
show ip static-route	Displays the configured static routes.

Configuration Examples 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
```

This example shows how to configure static routing for IPv6:

```
configure terminal
 ipv6 route 43::/64 42::2
 copy running-config startup-config
```

Additional References

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

- Related Documents, page 1-7

Related Documents

Related Topic	Document Title
Static Routing CLI	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Configuring Layer 3 Virtualization

This chapter describes how to configure Layer 3 virtualization.

This chapter includes the following sections:

Layer 3 Virtualization

This section includes the following topics:

Overview of Layer 3 Virtualization

Cisco NX-OS supports virtual routing and forwarding instances (VRFs). Each VRF contains a separate address space with unicast and multicast route tables for IPv4 and IPv6 and makes routing decisions independent of any other VRF.

Each router has a default VRF and a management VRF. All Layer 3 interfaces and routing protocols exist in the default VRF until you assign them to another VRF. The mgmt0 interface exists in the management VRF. With the VRF-lite feature, the switch supports multiple VRFs in customer edge (CE) switches. VRF-lite allows a service provider to support two or more Virtual Private Networks (VPNs) with overlapping IP addresses using one interface.

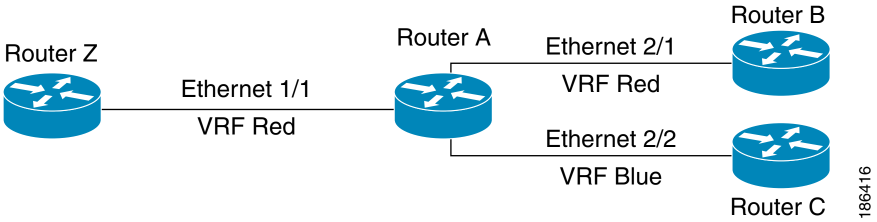
Note The switch does not use Multiprotocol Label Switching (MPLS) to support VPNs.

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. Figure 1-1 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 1-1 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 and export) between VRFs in a VRF lite scenario. The following are guidelines for the VRF route-leak feature:

- Supports route-leak between any two non-default VRFs and route-leak from the default VRF to any other VRF.
- Route-leak to the default VRF is not allowed as it is a global VRF.
- The route-leak feature is implemented using export and import route-targets under the VRF context.
- Filtering a part of the route-leak is done by using route-maps with the **match ip address** command.
- By default, the maximum prefix that can be leaked is 1000 routes. This is configurable.
- The route-leak feature must have an Enterprise license and the BGP feature enabled.

VRF-Lite

VRF-lite is a feature that enables a service provider to support two or more VPNs, where IP addresses can be overlapped among the VPNs. VRF-lite uses input interfaces to distinguish routes for different VPNs and forms virtual packet-forwarding tables by associating one or more Layer 3 interfaces with each VRF. Interfaces in a VRF can be either physical, such as Ethernet ports, or logical, such as VLAN SVIs, but a Layer 3 interface cannot belong to more than one VRF at any time.

Note Multiprotocol Label Switching (MPLS) and MPLS control plane are not supported in the VRF-lite implementation.

Note VRF-lite interfaces must be Layer 3 interfaces.

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 servics can select a particular VRF to reach a remote server or to filter information based on the selected VRF:

- AAA—See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*, for more information.
- Call Home—See the *Cisco Nexus 6000 Series NX-OS System Management Configuration Guide, Release 6.0*, for more information.
- HSRP—See Chapter 1, "Configuring HSRP" for more information.
- HTTP—See the *Cisco Nexus 6000 Series NX-OS Fundamentals Configuration Guide, Release 6.0*, for more information.
- Licensing—See the *Cisco NX-OS Licensing Guide* for more information.
- NTP—See the *Cisco Nexus 6000 Series NX-OS System Management Configuration Guide, Release 6.0*, for more information.
- RADIUS—See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*, for more information.
- Ping and Traceroute —See the *Cisco Nexus 6000 Series NX-OS Fundamentals Configuration Guide, Release 6.0*, for more information.
- SSH—See the *Cisco Nexus 6000 Series NX-OS Fundamentals Configuration Guide, Release 6.0*, for more information.
- SNMP—See the *Cisco Nexus 6000 Series NX-OS System Management Configuration Guide, Release 6.0*, for more information.
- Syslog—See the *Cisco Nexus 6000 Series NX-OS System Management Configuration Guide, Release 6.0*, for more information.
- TACACS+—See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*, for more information.
- TFTP—See the *Cisco Nexus 6000 Series NX-OS Fundamentals Configuration Guide, Release 6.0*, for more information.
- VRRP—See Chapter 1, "Configuring VRRP" for more information.

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

This section contains the following topics:

- Reachability, page 1-4
- Filtering, page 1-4

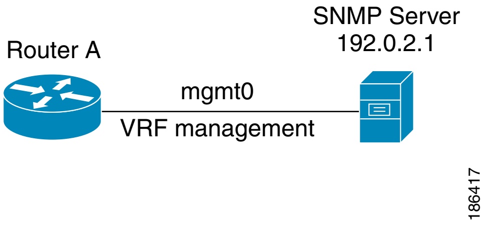
- Combining Reachability and Filtering, page 1-4

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 that Cisco NX-OS must use to reach the server.

Figure 1-2 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.

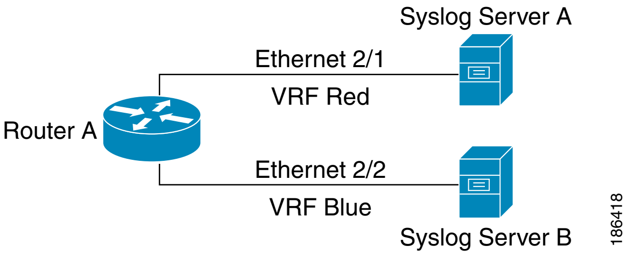
Figure 1-2 Service VRF Reachability



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. Figure 1-3 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 1-3 Service VRF Filtering

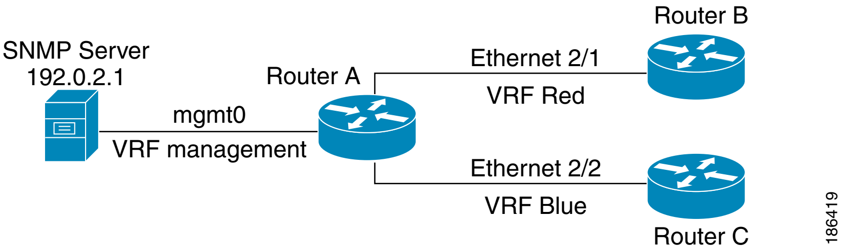


Combining Reachability and Filtering

You can combine reachability and filtering for VRF-aware services. You 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.

Figure 1-4 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 1-4 Service VRF Reachability Filtering



Licensing Requirements for VRFs

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	VRFs require no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*. **Note** The NX-OS base license allows you to use the default VRF and you can use the management VRF for the mgmt0 port. The two default VRFs are automatically created. VRF-lite allows you to create additional VRFs. The additional VRFs require the NX-OS base license as well.

Guidelines and Limitations

VRFs have the following configuration guidelines and limitations:

- When you make an interface a member of an existing VRF, Cisco NX-OS removes all Layer 3 configuration. 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.
- 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 configuration. You must use the **write erase** command and then the **write erase boot** command.

VRF-lite has the following guidelines and limitations:

- A switch with VRF-lite has a separate IP routing table for each VRF, which is separate from the global routing table.
- Because VRF-lite uses different VRF tables, the same IP addresses can be reused. Overlapped IP addresses are allowed in different VPNs.
- VRF-lite does not support all MPLS-VRF functionality; it does not support label exchange, LDP adjacency, or labeled packets.

- Multiple virtual Layer 3 interfaces can be connected to a VRF-lite switch.
- The switch supports configuring a VRF by using physical ports, VLAN SVIs, or a combination of both. The SVIs can be connected through an access port or a trunk port.
- The Layer 3 TCAM resource is shared between all VRFs.
- A switch using VRF can support one global network and up to 64 VRFs. The total number of routes supported is limited by the size of the TCAM.
- VRF-lite supports BGP, RIP, static routing, EIGRP, EIGRPv6, OSPF, and OSPFv3.
- VRF-lite does not affect the packet switching rate.

Default Settings

Table 1-1 lists the default settings for VRF parameters.

Table 1-1 Default VRF Parameters

Parameters	Default
Configured VRFs	default, management
routing context	default VRF

Configuring VRFs

This section contains the following topics:

- Creating a VRF, page 1-6
- Assigning VRF Membership to an Interface, page 1-8
- Configuring VRF Parameters for a Routing Protocol, page 1-9
- Configuring a VRF-Aware Service, page 1-11
- Setting the VRF Scope, page 1-12

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 in a switch.

SUMMARY STEPS

1. **configure terminal**
2. **vrf context** *name*
3. **ip route** {*ip-prefix* | *ip-addr ip-mask*} {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**tag** *tag-value* [*pref*]]

4. (Optional) **show vrf** [*vrf-name*]
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**vrf context** *name* **Example:** switch(config)# vrf context Enterprise 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.
Step 3	**ip route** {*ip-prefix* | *ip-addr ip-mask*} {[*next-hop* | *nh-prefix*] | [*interface next-hop* | *nh-prefix*]} [**tag** *tag-value* [*pref*] **Example**: 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	**show vrf** [*vrf-name*] **Example**: switch(config-vrf)# show vrf Enterprise	(Optional) Displays VRF information.
Step 5	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Saves this configuration change.

Use the **no vrf context** command to delete the VRF and the associated configuration:

Command	Purpose
no vrf context *name* **Example:** switch(config)# no vrf context Enterprise	Deletes the VRF and all associated configuration.

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

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

```
switch# configure terminal
switch(config)# vrf context Enterprise
switch(config-vrf)# ip route 192.0.2.0/8 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.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrf member** *vrf-name*
5. **ip-address** *ip-prefix/length*
6. (Optional) **show vrf** *vrf-name* **interface** *interface-type number*
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`interface-type slot/port`*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot*/*port* syntax is *slot*/*QSFP-module*/*port*.

Step 3	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	**`vrf member`** *`vrf-name`*
	Example:
	`switch(config-if)# vrf member RemoteOfficeVRF`
		Adds this interface to a VRF.

Step 5	**`ip address`** *`ip-prefix/length`*
	Example:
	`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.

	Command	Purpose
Step 6	**show vrf** *vrf-name* **interface** *interface-type number* **Example**: `switch(config-vrf)# show vrf Enterprise interface ethernet 1/2`	(Optional) Displays VRF information.
Step 7	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

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

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
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.

SUMMARY STEPS

1. **configure terminal**
2. **router ospf** *instance-tag*
3. **vrf** *vrf-name*
4. (Optional) **maximum-paths** *paths*
5. **interface** *interface-type slot/port*
6. **no switchport**
7. **vrf member** *vrf-name*
8. **ip address** *ip-prefix/length*
9. **ip router ospf** i*nstance-tag* **area** *area-id*
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**router ospf** *instance-tag* **Example:** switch(config-vrf)# router ospf 201 switch(config-router)#	Creates a new OSPFv2 instance with the configured instance tag.
Step 3	**vrf** *vrf-name* **Example:** switch(config-router)# vrf RemoteOfficeVRF switch(config-router-vrf)#	Enters VRF configuration mode.
Step 4	**maximum-paths** *paths* Example: switch(config-router-vrf)# maximum-paths 4	(Optional) Configures the maximum number of equal OSPFv2 paths to a destination in the route table for this VRF. Used for load balancing.
Step 5	**interface** *interface-type slot/port* **Example**: switch(config)# interface ethernet 1/2 switch(config-if)#	Enters interface configuration mode. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 6	**no switchport** **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 7	**vrf member** *vrf-name* **Example:** switch(config-if)# vrf member RemoteOfficeVRF	Adds this interface to a VRF.
Step 8	**ip address** *ip-prefix/length* **Example:** 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* **Example:** switch(config-if)# ip router ospf 201 area 0	Assigns this interface to the OSPFv2 instance and area configured.
Step 10	**copy running-config startup-config** **Example:** switch(config)# copy running-config startup-config	(Optional) Saves this configuration change.

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)# no switchport
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. See the "VRF-Aware Services" section on page 1-3 for links to the appropriate chapter or configuration guide for information on how to configure the service for VRFs. This section uses SNMP and IP domain lists as example services for the detailed configuration steps.

SUMMARY STEPS

1. **configure terminal**
2. **snmp-server host** *ip-address* [**filter_vrf** *vrf-name*] [**use-vrf** *vrf-name*]
3. **vrf context** [*vrf-name*]
4. **ip domain-list** *domain-name* [**all-vrfs**] [**use-vrf** *vrf-name*]
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`snmp-server host`** *`ip-address`* `[`**`filter-vrf`** *`vrf-name`*`] [`**`use-vrf`** *`vrf-name`*`]`
	
	Example:
	`switch(config)# snmp-server host 192.0.2.1 use-vrf Red`
	`switch(config-vrf)#`
		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 3	**`vrf context`** *`vrf-name`*
	
	Example:
	`switch(config)# vrf context Blue`
	`switch(config-vrf)#`
		Creates a new VRF.

	Command	Purpose

Step 4	**ip domain-list** *domain-name* [**all-vrfs**][**use-vrf** *vrf-name*]
	Example:
	`switch(config-vrf)# ip domain-list List all-vrfs use-vrf Blue`
	`switch(config-vrf)#`
		Configures the domain list in the VRF and optionally configures the VRF that Cisco NX-OS uses to reach the domain name listed.

Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

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 definition 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). This 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.

To set the VRF scope, use the following command in EXEC mode:

Command	Purpose

routing-context vrf *vrf-name*
Example:
`switch# routing-context vrf red`
`switch%red#`
	Sets the routing context for all EXEC commands. Default routing context is the default VRF.

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

Command	Purpose

routing-context vrf default
Example:
`switch%red# routing-context vrf default`
`switch#`
	Sets the default routing context.

Verifying the VRF Configuration

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

Command	Purpose
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. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Configuration Examples for VRF

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

```
configure terminal
vrf context Red
 snmp-server host 192.0.2.12 use-vrf Red
router ospf 201
interface ethernet 1/2
 no switchport
 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.:

```
configure terminal
!Create the VRFs
vrf context Red
vrf context Blue
!Create the OSPF instances and associate them with each VRF
feature ospf
router ospf Lab
 vrf Red
router ospf Production
 vrf Blue
!Configure one interface to use ospf Lab on VRF Red
interface ethernet 1/2
 no switchport
 vrf member Red
 ip address 192.0.2.1/16
 ip router ospf Lab area 0
 no shutdown
!Configure another interface to use ospf Production on VRF Blue
interface ethernet 10/2
 no switchport
 vrf member Blue
 ip address 192.0.2.1/16
 ip router ospf Production area 0
 no shutdown
```

```
!configure the SNMP server
snmp-server user admin network-admin auth md5 nbv-12345
snmp-server community public ro
!Create the SNMP contexts for each VRF
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.
```

Related Topics

The following topics can give more information on VRFs:

- *Cisco Nexus 6000 Series NX-OS Fundamentals Configuration Guide, Release 6.0*
- *Cisco Nexus 6000 Series NX-OS System Management Configuration Guide, Release 6.0*

Additional References

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

- Related Documents, page 1-14
- Standards, page 1-14

Related Documents

Related Topic	Document Title
VRF CLI	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

CHAPTER 1

Managing the Unicast RIB and FIB

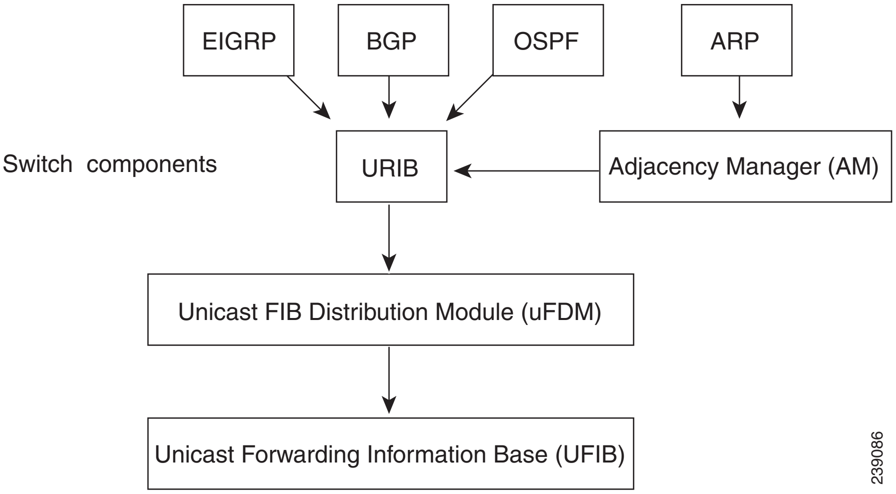
This chapter describes how to manage routes in the unicast Routing Information Base (RIB) and the Forwarding Information Base (FIB) on the Cisco NX-OS switch.

This chapter includes the following sections:

Information About the Unicast RIB and FIB

The unicast RIB (IPv4 RIB) and FIB are part of the Cisco NX-OS forwarding architecture, as shown in Figure 1-1.

Figure 1-1 Cisco NX-OS Forwarding Architecture


EIGRP
BGP
OSPF
ARP
Switch components
URIB
Adjacency Manager (AM)
Unicast FIB Distribution Module (uFDM)
Unicast Forwarding Information Base (UFIB)
239086

The unicast RIB 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 unicast forwarding information base (FIBs) 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).

This section includes the following topic:

- FIB Tables, page 1-2

FIB Tables

The hardware provides two tables: a TCAM table and a hash table. The TCAM table is shared between longest prefix match (LPM) route /32 unicast route. The hash table is shared between the /32 unicast entries and the multicast entries. The LPM table has a maximum of 32000 routes and the hash table has a maximum of 128000 routes.

Licensing Requirements for the Unicast RIB and FIB

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	The unicast RIB and FIB require no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Managing the Unicast RIB and FIB

This section includes the following topics:

- Displaying Module FIB Information, page 1-3
- Configuring Load Sharing in the Unicast FIB, page 1-3Displaying Routing and Adjacency Information, page 1-4
- Clearing Forwarding Information in the FIB, page 1-5Estimating Memory Requirements for Routes, page 1-6
- Clearing Routes in the Unicast RIB, page 1-6

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.

Displaying Module FIB Information

You can display the FIB information on a switch.

DETAILED STEPS

To display the FIB information on a switch, use the following commands in any mode:

Command	Purpose
show ip fib adjacency { **Example:** switch# show ip fib adjacency	Displays the adjacency information for FIB.
show forwarding {**ipv4 adjacency** { **Example:** switch# show forwarding ipv4 adjacency	Displays the adjacency information for IPv4.
show ip fib interfaces **Example:** switch# show ip fib interfaces	Displays the FIB interface information for IPv4.
show ip fib route adjacency { **Example:** switch# show ip fib route	Displays the route table for IPv4.
show forwarding ipv4 route adjacency { **Example:** switch# show forwarding ipv4 route	Displays the route table for IPv4.

This example shows how to display the FIB contents on a switch:

```
switch# show ip fib route

IPv4 routes for table default/base

------------------+------------------+---------------------
Prefix            | Next-hop         | Interface
------------------+------------------+---------------------
0.0.0.0/32          Drop               Null0
255.255.255.255/32  Receive            sup-eth1
```

Configuring Load Sharing in the Unicast FIB

Dynamic routing protocols, such as Open Shortest Path First (OSPF), support load balancing with equal-cost multipath (ECMP). The routing protocol determines its best routes based on the metrics configured for the protocol and installs up to the protocol-configured maximum paths in the unicast RIB. The unicast RIB compares the administrative distances of all routing protocol paths in the RIB and selects a best path set from all of the path sets installed by the routing protocols. The unicast RIB installs this best path set into the FIB for use by the forwarding plane.

The forwarding plane uses a load-sharing algorithm to select one of the installed paths in the FIB to use for a given data packet.

You can globally configure the following load-sharing settings:

- load-share mode—Selects the best path based on the destination address and port or the source and the destination address and port.
- Universal ID—Sets the random seed for the hash algorithm. You do not need to configure the Universal ID. Cisco NX-OS chooses the Universal ID if you do not configure it.

Note Load sharing uses the same path for all packets in a given flow. A flow is defined by the load-sharing method that you configure. For example, if you configure source-destination load sharing, then all packets with the same source IP address and destination IP address pair follow the same path.

To configure the unicast FIB load-sharing algorithm, use the following command in global configuration mode:

Command	Purpose
ip load-sharing address {**destination port destination** | **source-destination** [**port source-destination**]} [**universal-id** *seed*] **Example:** `switch(config)# ip load-sharing address source-destination`	Configures the unicast FIB load-sharing algorithm for data traffic. The *universal-id* range is from 1 to 4294967295.

To display the unicast FIB load-sharing algorithm, use the following command in any mode:

Command	Purpose
show ip load-sharing **Example:** `switch(config)# show ip load-sharing`	Displays the unicast FIB load-sharing algorithm for data traffic.

This example shows how to display the route selected for a source/destination pair:

```
switch# show routing hash 10.0.0.5 30.0.0.2
  Load-share parameters used for software forwarding:
  load-share mode: address source-destination port source-destination
  Universal-id seed: 0xe05e2e85
  Hash for VRF "default"
  Hashing to path *20.0.0.2 (hash: 0x0e), for route:
```

Displaying Routing and Adjacency Information

You can display the routing and adjacency information.

To display the routing and adjacency information, use the following commands in any mode:

Command	Purpose
show ip route [*route-type* | **interface** *int-type number* | **next-hop**] **Example:** switch# show ip route	Displays the unicast route table. The *route-type* argument can be a single route prefix, direct, static, or a dynamic route protocol. Use the **?** keyword to see the supported interfaces.
show ip adjacency [*prefix* | *interface-type number* [**summary**]| **non-best**] [**detail**] [**vrf** *vrf-id*] **Example:** switch# show ip adjacency	Displays the adjacency table. The argument ranges are as follows: • *prefix*—Any IPv4 prefix address. • *interface-type number*—Use the **?** keyword to see the supported interfaces. • *vrf-id*—Any case-sensitive, alphanumeric string up to 32 characters.
show ip routing [*route-type* | **interface** *int-type number* | **next-hop** | **recursive-next-hop** | **summary** | **updated** {**since** | **until**} *time*] **Example:** switch# show routing summary	Displays the unicast route table. The *route-type* argument can be a single route prefix, direct, static, or a dynamic route protocol. Use the **?** keyword to see the supported interfaces.

This example shows how to display the unicast route table:

```
switch# show ip route
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]

192.168.0.2/24, ubest/mbest: 1/0, attached
    *via 192.168.0.32, Eth1/5, [0/0], 22:34:09, direct
192.168.0.32/32, ubest/mbest: 1/0, attached
    *via 192.168.0.32, Eth1/5, [0/0], 22:34:09, local
```

This example shows the adjacency information:

```
switch# show ip adjacency

IP Adjacency Table for VRF default
Total number of entries: 2
Address         Age       MAC Address     Pref Source     Interface       Best
10.1.1.1        02:20:54  00e0.b06a.71eb  50   arp        mgmt0           Yes
10.1.1.253      00:06:27  0014.5e0b.81d1  50   arp        mgmt0           Yes
```

Clearing Forwarding Information in the FIB

You can clear one or more entries in the FIB. Clearing a FIB entry does not affect the unicast RIB.

Caution The **clear forwarding** command disrupts forwarding on the switch.

To clear an entry in the FIB, including a Layer 3 inconsistency, use the following command in any mode:

Estimating Memory Requirements for Routes

Command	Purpose
clear forwarding {**ipv4**} **route** {* | *prefix*} [**vrf** *vrf-name*] [**module** {*slot*| **all**}] **Example:** switch(config)# clear forwarding ipv4 route *	Clears one or more entries from the FIB. The route options are as follows: • *—All routes. • *prefix*—Any IP prefix. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters. The *slot* range is from 1 to 10.

You can estimate the memory that a number of routes and next-hop addresses will use.

To estimate the memory requirements for routes, use the following command in any mode:

Command	Purpose
show routing memory estimate routes *num-routes* **next-hops** *num-nexthops* **Example:** switch# show routing memory estimate routes 1000 next-hops 1	Displays the memory requirements for routes. The *num-routes* range is from 1000 to 1000000. The *num-nexthops* range is from 1 to 16.

Clearing Routes in the Unicast RIB

You can clear one or more routes from the unicast RIB.

Caution The * keyword is severely disruptive to routing.

To clear one or more entries in the unicast RIB, use the following commands in any mode:

Command	Purpose
clear ip route {* | {*route* | *prefix/length*}[*next-hop interface*]} [**vrf** *vrf-name*] **Example:** `switch(config)# clear ip route 10.2.2.2`	Clears one or more routes from both the unicast RIB and all the module FIBs. The route options are as follows: • *—All routes. • *route*—An individual IP route. • *prefix/length*—Any IP prefix. • *next-hop*—The next-hop address • *interface*—The interface to reach the next-hop address. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.
clear routing [**multicast** | **unicast**] [**ip** | **ipv4**] {* | {*route* | *prefix/length*}[*next-hop interface*]} [**vrf** *vrf-name*] **Example:** `switch(config)# clear routing ip 10.2.2.2`	Clears one or more routes from the unicast RIB. The route options are as follows: • *—All routes. • *route*—An individual IP route. • *prefix/length*—Any IP prefix. • *next-hop*—The next-hop address • *interface*—The interface to reach the next-hop address. The *vrf-name* can be any case-sensitive, alphanumeric string up to 32 characters.

Verifying the Unicast RIB and FIB Configuration

To display the unicast RIB and FIB configuration information, perform one of the following tasks:

Command	Purpose
show forwarding adjacency	Displays the adjacency table on a module.
show forwarding distribution {**clients** | **fib-state**}	Displays the FIB distribution information.
show forwarding interfaces	Displays the FIB information for a interface.
show forwarding ipv4 route	Displays routes in the FIB.
show ip adjacency	Displays the adjacency table.
show ip route	Displays IPv4 routes from the unicast RIB.
show routing	Displays routes from the unicast RIB.

Additional References

For additional information related to managing unicast RIB and FIB, see the following sections:

- Related Documents, page 1-8

Related Documents

Related Topic	**Document Title**
Unicast RIB and FIB CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Configuring Route Policy Manager

This chapter describes how to configure the Route Policy Manager on the Cisco NX-OS switch.

This chapter includes the following sections:

Information About Route Policy Manager

Route Policy Manager supports route maps and IP prefix lists. These features are used for route redistribution. A prefix list contains one or more IPv4 network prefixes and the associated prefix length values. You can use a prefix list by itself in features such as Border Gateway Protocol (BGP) templates, route filtering, or redistribution of routes that are exchanged between routing domains.

Route maps can apply to both routes and IP packets. Route filtering and redistribution pass a route through a route map.

This section includes the following topics:

Prefix Lists

You can use prefix lists to permit or deny an address or range of addresses. Filtering by a prefix list involves matching the prefixes of routes or packets with the prefixes listed in the prefix list. An implicit deny is assumed if a given prefix does not match any entries in a prefix list.

You can configure multiple entries in a prefix list and permit or deny the prefixes that match the entry. Each entry has an associated sequence number that you can configure. If you do not configure a sequence number, Cisco NX-OS assigns a sequence number automatically. Cisco NX-OS evaluates prefix lists starting with the lowest sequence number. Cisco NX-OS processes the first successful match for a given prefix. Once a match occurs, Cisco NX-OS processes the permit or deny statement and does not evaluate the rest of the prefix list.

Note An empty prefix list permits all routes.

MAC Lists

You can use MAC lists to permit or deny MAC address or range of addresses. A MAC list consists of a list of MAC addresses and optional MAC masks. A MAC mask is a wild-card mask that is logically AND-ed with the MAC address when the route map matches on the MAC list entry. Filtering by a MAC list involves matching the MAC address of packets with the MAC addresses listed in the MAC list. An implicit deny is assumed if a given MAC address does not match any entries in a MAC list.

You can configure multiple entries in a MAC list and permit or deny the MAC addresses that match the entry. Each entry has an associated sequence number that you can configure. If you do not configure a sequence number, Cisco NX-OS assigns a sequence number automatically. Cisco NX-OS evaluates MAC lists starting with the lowest sequence number. Cisco NX-OS processes the first successful match for a given MAC address. Once a match occurs, Cisco NX-OS processes the permit or deny statement and does not evaluate the rest of the MAC list.

Route Maps

You can use route maps for route redistribution. Route map entries consist of a list of match and set criteria. The match criteria specify match conditions for incoming routes or packets, and the set criteria specify the action taken if the match criteria are met.

You can configure multiple entries in the same route map. These entries contain the same route map name and are differentiated by a sequence number.

You create a route map with one or more route map entries arranged by the sequence number under a unique route map name. The route map entry has the following parameters:

- Sequence number
- Permission—permit or deny
- Match criteria
- Set changes

By default, a route map processes routes or IP packets in a linear fashion, that is, starting from the lowest sequence number. You can configure the route map to process in a different order using the **continue** statement, which allows you to determine which route map entry to process next.

Match Criteria

You can use a variety of criteria to match a route or IP packet in a route map. Some criteria, such as BGP community lists, are applicable only to a specific routing protocol, while other criteria, such as the IP source or the destination address, can be used for any route or IP packet.

When Cisco NX-OS processes a route or packet through a route map, it compares the route or packet to each of the match statements configured. If the route or packet matches the configured criteria, Cisco NX-OS processes it based on the permit or deny configuration for that match entry in the route map and any set criteria configured.

The match categories and parameters are as follows:

- BGP parameters—Match based on AS numbers, AS-path, community attributes, or extended community attributes.
- Prefix lists—Match based on an address or range of addresses.
- Multicast parameters—Match based on rendezvous point, groups, or sources.
- Other parameters—Match based on IP next-hop address or packet length.

Set Changes

Once a route or packet matches an entry in a route map, the route or packet can be changed based on one or more configured set statements.

The set changes are as follows:

- BGP parameters—Change the AS-path, tag, community, extended community, dampening, local preference, origin, or weight attributes.
- Metrics—Change the route-metric, the route-tag, or the route-type.
- Other parameters—Change the forwarding address or the IP next-hop address.

Access Lists

IP access lists can match the packet to a number of IP packet fields such as the following:

- Source or destination IPv4 or IPv6 address
- Protocol
- Precedence
- ToS

See the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*, for more information on ACLs.

AS Numbers for BGP

You can configure a list of AS numbers to match against BGP peers. If a BGP peer matches an AS number in the list and matches the other BGP peer configuration, BGP creates a session. If the BGP peer does not match an AS number in the list, BGP ignores the peer. You can configure the AS numbers as a list, a range of AS numbers, or you can use an AS-path list to compare the AS numbers against a regular expression.

AS-path Lists for BGP

You can configure an AS-path list to filter inbound or outbound BGP route updates. If the route update contains an AS-path attribute that matches an entry in the AS-path list, the router processes the route based on the permit or deny condition configured. You can configure AS-path lists within a route map.

You can configure multiple AS-path entries in an AS-path list by using the same AS-path list name. The router processes the first entry that matches.

Community Lists for BGP

You can filter BGP route updates based on the BGP community attribute by using community lists in a route map. You can match the community attribute based on a community list, and you can set the community attribute using a route map.

A community list contains one or more community attributes. If you configure more than one community attribute in the same community list entry, then the BGP route must match all community attributes listed to be considered a match.

You can also configure multiple community attributes as individual entries in the community list by using the same community list name. In this case, the router processes the first community attribute that matches the BGP route, using the permit or deny configuration for that entry.

You can configure community attributes in the community list in one of the following formats:

- A named community attribute, such as **internet** or **no-export.**
- In *aa:nn* format, where the first two bytes represent the two-byte AS number and the last two bytes represent a user-defined network number.
- A regular expression.

See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for more information on regular expressions.

Extended Community Lists for BGP

Extended community lists support 4-byte AS numbers. You can configure community attributes in the extended community list in one of the following formats:

- In *aa4:nn* format, where the first four bytes represent the four-byte AS number and the last two bytes represent a a user-defined network number.
- A regular expression.

See the *Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*, for more information on regular expressions.

Cisco NX-OS supports generic-specific extended community lists, which provide similar functionality to regular community lists for four-byte AS numbers. You can configure generic-specific extended community lists with the following properties:

- Transitive—BGP propagates the community attributes across autonomous systems.
- Nontransitive—BGP removes community attributes before propagating the route to another autonomous system.

Route Redistribution and Route Maps

You can use route maps to control the redistribution of routes between routing domains. Route maps match on the attributes of the routes to redistribute only those routes that pass the match criteria. The route map can also modify the route attributes during this redistribution using the set changes.

The router matches redistributed routes against each route map entry. If there are multiple match statements, the route must pass all of the match criteria. If a route passes the match criteria defined in a route map entry, the actions defined in the entry are executed. If the route does not match the criteria, the router compares the route against subsequent route map entries. Route processing continues until a match is made or the route is processed by all entries in the route map with no match. If the router processes the route against all entries in a route map with no match, the router accepts the route (inbound route maps) or forwards the route (outbound route maps).

Licensing Requirements for Route Policy Manager

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Route Policy Manager requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations

Route Policy Manager has the following configuration guidelines and limitations:

- An empty route map denies all the routes.
- An empty prefix list permits all the routes.
- Without any match statement in a route-map entry, the permission (permit or deny) of the route-map entry decides the result for all the routes or packets.
- If referred policies (for example, prefix lists) within a match statement of a route-map entry return either a no-match or a deny-match, Cisco NX-OS fails the match statement and processes the next route-map entry.
- When you change a route map, Cisco NX-OS holds all the changes until you exit from the route-map configuration submode. Cisco NX-OS then sends all the changes to the protocol clients to take effect.
- Because you can use a route map before you define it, verify that all your route maps exist when you finish a configuration change.
- You can view the route-map usage for redistribution and filtering. Each individual routing protocol provides a way to display these statistics.

Default Settings

Table 1-1 lists the default settings for Route Policy Manager.

Table 1-1 Default Route Policy Manager Parameters

Parameters	Default
Route Policy Manager	Enabled

Configuring Route Policy Manager

Route Policy Manager configuration includes the following topics:

- Configuring IP Prefix Lists, page 1-6
- Configuring MAC Lists, page 1-8
- Configuring AS-path Lists, page 1-8
- Configuring Community Lists, page 1-9
- Configuring Extended Community Lists, page 1-11
- Configuring Route Maps, page 1-12

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 IP Prefix Lists

IP prefix lists match the IP packet or route against a list of prefixes and prefix lengths. You can create an IP prefix list for IPv4 and create an IPv6 prefix list for IPv6.

You can configure the prefix list entry to match the prefix length exactly, or to match any prefix with a length that matches the configured range of prefix lengths.

Use the **ge** and **lt** keywords to create a range of possible prefix lengths. The incoming packet or route matches the prefix list if the prefix matches and if the prefix length is greater than or equal to the **ge** keyword value (if configured) and less than or equal to the **lt** keyword value (if configured).

SUMMARY STEPS

1. **configure terminal**
2. (Optional) {**ip** | **ipv6**} **prefix-list** *name* **description** *string*
3. **ip prefix-list** *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}]

   or

   ipv6 prefix-list *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}]
4. (Optional) **show** {**ip** | **ipv6**} **prefix-list** *name*
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	{**ip** | **ipv6**} **prefix-list** *name* **description** *string* **Example:** `switch(config)# ip prefix-list AllowPrefix description allows engineering server`	(Optional) Adds an information string about the prefix list.
Step 3	**ip prefix-list** *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] **Example:** `switch(config)# ip prefix-list AllowPrefix seq 10 permit 192.0.2.0 eq 24`	Creates an IPv4 prefix list or adds a prefix to an existing prefix list. The prefix length is matched as follows: • eq—Matches the exact *prefix length*. • ge—Matches a prefix length that is equal to or greater than the configured *prefix length*. • le—Matches a prefix length that is equal to or less than the configured *prefix length*.
Step 4	**ip prefix-list** *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] **Example:** `switch(config)# ip prefix-list AllowPrefix seq 10 permit 192.0.2.0 eq 24`	Creates an IPv4 prefix list or adds a prefix to an existing prefix list. The prefix length is matched as follows: • eq—Matches the exact *prefix length*. • ge—Matches a prefix length that is equal to or greater than the configured *prefix length*. • le—Matches a prefix length that is equal to or less than the configured *prefix length*.
	ipv6 prefix-list *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] **Example:** `switch(config)# ipv6 prefix-list AllowIPv6Prefix seq 10 permit 2001:0DB8:: le 32`	Creates an IPv6 prefix list or adds a prefix to an existing prefix list. The prefix length is configured as follows: • eq—Matches the exact *prefix length*. • ge—Matches a prefix length that is equal to or greater than the configured *prefix length*. • le—Matches a prefix length that is equal to or less than the configured *prefix length*.
Step 5	**show** {**ip** | **ipv6**} **prefix-list** *name* **Example:** `switch(config)# show ip prefix-list AllowPrefix`	(Optional) Displays information about prefix lists.
Step 6	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to create an IPv4 prefix list with two entries and apply the prefix list to a BGP neighbor:

```
switch# configure terminal
switch(config)# ip prefix-list allowprefix seq 10 permit 192.0.2.0/24 eq 24
switch(config)# ip prefix-list allowprefix seq 20 permit 209.165.201.0/27 eq 27
switch(config)# router bgp 65536:20
switch(config-router)# neighbor 192.0.2.1/16 remote-as 65535:20
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# prefix-list allowprefix in
```

Configuring MAC Lists

You can configure a MAC list to permit or deny a range of MAC addresses.

SUMMARY STEPS

1. **configure terminal**
2. **mac-list** *name* **[seq** *number***] {permit | deny}** *mac-address* **[***mac-mask***]**
3. (Optional) **show mac-list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**mac-list** *name* [**seq** *number*] {**permit** | **deny**} *mac-address* {mac-mask] **Example:** switch(config)# mac-list AllowMac seq 1 permit 0022.5579.a4c1 ffff.ffff.0000	Creates a MAC list or adds a MAC address to an existing MAC list. The *seq* range is from 1 to 4294967294. The *mac-mask* specifies the portion of the MAC address to match against and is in MAC address format.
Step 3	**show mac-list** *name* **Example:** switch(config)# show mac-list AllowMac	(Optional) Displays information about MAC lists.
Step 4	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

Configuring AS-path Lists

You can specify an AS-path list filter on both inbound and outbound BGP routes. Each filter is an access list based on regular expressions. If the regular expression matches the representation of the AS-path attribute of the route as an ASCII string, then the permit or deny condition applies.

SUMMARY STEPS

1. **configure terminal**
2. **ip as-path access-list** *name* {**deny** | **permit**} *expression*
3. (Optional) **show ip as-path list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**ip as-path access-list** *name* {**deny** | **permit**} *expression* **Example:** switch(config)# ip as-path access-list Allow40 permit 40	Creates a BGP AS-path list using a regular expression.
Step 3	**show** {**ip** | **ipv6**} **as-path-access-list** *name* **Example**: switch(config)# show ip as-path-access-list Allow40	(Optional) Displays information about as-path access lists.
Step 4	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to create an AS-path list with two entries and apply the AS path list to a BGP neighbor:

```
switch# configure terminal
switch(config)# ip as-path access-list AllowAS permit 64510
switch(config)# ip as-path access-list AllowAS permit 64496
switch(config)# copy running-config startup-config
switch(config)# router bgp 65536:20
switch(config-router)# neighbor 192.0.2.1/16 remote-as 65535:20
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# filter-list AllowAS in
```

Configuring Community Lists

You can use community lists to filter BGP routes based on the community attribute. The community number consists of a 4-byte value in the *aa:nn* format. The first two bytes represent the autonomous system number, and the last two bytes represent a user-defined network number.

When you configure multiple values in the same community list statement, all community values must match to satisfy the community list filter. When you configure multiple values in separate community list statements, the first list that matches a condition is processed.

Use community lists in a match statement to filter BGP routes based on the community attribute.

SUMMARY STEPS

1. **configure terminal**
2. **ip community-list standard** *list-name* {**deny** | **permit**} [*community-list*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]
   or
   ip community-list expanded *list-name* {**deny** | **permit**} *expression*
3. (Optional) **show ip community-list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**ip community-list standard** *list-name* {**deny** | **permit**} [*community-list*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**] **Example:** switch(config)# ip community-list standard BGPCommunity permit no-advertise 65536:20	Creates a standard BGP community list. The *list-name* can be any case-sensitive, alphanumeric string up to 63 characters. The *community-list* can be one or more communities in the *aa:nn* format.
	ip community-list expanded *list-name* {**deny** | **permit**} *expression* **Example:** switch(config)# ip community-list expanded BGPComplex deny 50000:[0-9][0-9]_	Creates an expanded BGP community list using a regular expression.
Step 3	**show ip community-list** *name* **Example**: switch(config)# show ip community-list BGPCommunity	(Optional) Displays information about community lists.
Step 4	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to create a community list with two entries:

```
switch# configure terminal
switch(config)# ip community-list standard BGPCommunity permit no-advertise 65536:20
switch(config)# ip community-list standard BGPCommunity permit local-AS no-export
switch(config)# copy running-config startup-config
```

Configuring Extended Community Lists

You can use extended community lists to filter BGP routes based on the community attribute. The community number consists of a 6-byte value in the *aa4:nn* format. The first four bytes represent the autonomous system number, and the last two bytes represent a user-defined network number.

When you configure multiple values in the same extended community list statement, all extended community values must match to satisfy the extended community list filter. When you configure multiple values in separate extended community list statements, the first list that matches a condition is processed.

Use extended community lists in a match statement to filter BGP routes based on the extended community attribute.

SUMMARY STEPS

1. **configure terminal**
2. **ip extcommunity-list standard** *list-name* {**deny** | **permit**} **4bytegeneric** {**transitive** | **non-transitive**} *community1* [*community2*]
   ip extcommunity-list expanded *list-name* {**deny** | **permit**} *expression*
3. (Optional) **show ip extcommunity-list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**ip extcommunity-list standard** *list-name* {**deny** | **permit**} **4bytegeneric** {**transitive** | **nontransitive**} *community1* [*community2...*] **Example:** `switch(config)# ip extcommunity-list standard BGPExtCommunity permit 4bytegeneric transitive 65536:20`	Creates a standard BGP extended community list. The *community* can be one or more extended communities in the *aa4:nn* format.
	ip extcommunity-list expanded *list-name* {**deny** | **permit**} *expression* **Example:** `switch(config)# ip extcommunity-list expanded BGPExtComplex deny 1.5:[0-9][0-9]_`	Creates an expanded BGP extended community list using a regular expression.

	Command	Purpose

Step 3	**`show ip community-list`** *`name`*
	Example:
	`switch(config)# show ip community-list BGPCommunity`
		(Optional) Displays information about extended community lists.

Step 4	**`copy running-config startup-config`**
	Example:
	`switch# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to create a generic-specific extended community list:

```
switch# configure terminal
switch(config)# ip extcommunity-list standard test1 permit 4bytegeneric transitive
65536:40 65536:60
switch(config)# copy running-config startup-config
```

Configuring Route Maps

You can use route maps for route redistribution or route filtering. Route maps can contain multiple match criteria and multiple set criteria.

Configuring a route map for BGP triggers an automatic soft clear or refresh of BGP neighbor sessions.

SUMMARY STEPS

1. **configure terminal**
2. **route-map** *map-name* [**permit** | **deny**] [*seq*]
3. (Optional) **continue** *seq*
4. (Optional) **exit**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`route-map`** *`map-name`* `[`**`permit`** `|` **`deny`**`]` `[`*`seq`*`]`
	Example:
	`switch(config)# route-map Testmap permit 10`
	`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.

Step 3	**`continue`** *`seq`*
	Example:
	`switch(config-route-map)# continue 10`
		(Optional) Determines what sequence statement to process next in the route map. Used only for filtering and redistribution.

	Command	Purpose

Step 4	**`exit`**
	`Example:`
	`switch(config-route-map)# exit`
		(Optional) Exits route-map configuration mode.

Step 5	**`copy running-config startup-config`**
	`Example:`
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

You can configure the following optional match parameters for route maps in route-map configuration mode:

Note The **default-information originate** command ignores **match** statements in the optional route map.

Command	Purpose

`match as-path` *`name`* `[`*`name...`*`]`
`Example:`
`switch(config-route-map)# match as-path Allow40`
	Matches against one or more AS-path lists. Create the AS-path list with the **ip as-path access-list** command.

`match as-number` `{`*`number`* `[,`*`number...`*`] |` **`as-path-list`** *`name`* `[`*`name...`*`]}`
`Example:`
`switch(config-route-map)# match as-number 33,50-60`
	Matches against one or more AS numbers or AS-path lists. Create the AS-path list with the **ip as-path access-list** command. The number range is from 1 to 65535. The AS-path list name can be any case-sensitive, alphanumeric string up to 63 characters.

`match community` *`name`* `[`*`name...`*`][`***`exact-match`***`]`
`Example:`
`switch(config-route-map)# match community BGPCommunity`
	Matches against one or more community lists. Create the community list with the **ip community-list** command.

`match extcommunity` *`name`* `[`*`name...`*`][`***`exact-match`***`]`
`Example:`
`switch(config-route-map)# match extcommunity BGPextCommunity`
	Matches against one or more extended community lists. Create the community list with the **ip extcommunity-list** command.

`match interface` *`interface-type number`* `[`*`interface-type number...`*`]`
`Example:`
`switch(config-route-map)# match interface e 1/2`
	Matches any routes that have their next hop out one of the configured interfaces. Use **?** to find a list of supported interface types.

`match ip address prefix-list` *`name`* `[`*`name...`*`]`
`Example:`
`switch(config-route-map)# match ip address prefix-list AllowPrefix`
	Matches against one or more IPv4 prefix lists. Use the **ip prefix-list** command to create the prefix list.

Command	Purpose
match ipv6 address prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip address prefix-list AllowIPv6Prefix`	Matches against one or more IPv6 prefix lists. Use the **ipv6 prefix-list** command to create the prefix list.
match ip multicast [**source** *ipsource*] [[**group** *ipgroup*] [*rp iprp*]] **Example:** `switch(config-route-map)# match ip multicast rp 192.0.2.1`	Matches an IPv4 multicast packet based on the multicast source, group, or rendezvous point.
match ipv6 multicast [**source** *ipsource*] [[**group** *ipgroup*] [*rp iprp*]] **Example:** `switch(config-route-map)# match ip multicast source 2001:0DB8::1`	Matches an IPv6 multicast packet based on the multicast source, group, or rendezvous point.
match ip next-hop prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip next-hop prefix-list AllowPrefix`	Matches the IPv4 next-hop address of a route to one or more IP prefix lists. Use the **ip prefix-list** command to create the prefix list.
match ipv6 next-hop prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ipv6 next-hop prefix-list AllowIPv6Prefix`	Matches the IPv6 next-hop address of a route to one or more IP prefix lists. Use the **ipv6 prefix-list** command to create the prefix list.
match ip route-source prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip route-source prefix-list AllowPrefix`	Matches the IPv4 route source address of a route to one or more IP prefix lists. Use the **ip prefix-list** command to create the prefix list.
match ipv6 route-source prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ipv6 route-source prefix-list AllowIPv6Prefix`	Matches the IPv6 route-source address of a route to one or more IP prefix lists. Use the **ipv6 prefix-list** command to create the prefix list.
match mac-list *name* [*name...*] **Example:** `switch(config-route-map)# match mac-list AllowMAC`	Matches against one or more MAC lists. Use the **mac-list** command to create the MAC list.
match metric *value* [**+-** *deviation.*] [*value..*] **Example:** `switch(config-route-map)# match mac-list AllowMAC`	Matches the route metric against one or more metric values or value ranges. Use +- *deviation* argument to set a metric range. The route map matches any route metric that falls the range: *value* - *deviation* to *value* + *deviation*.

Command	Purpose
match route-type *route-type* **Example:** `switch(config-route-map)# match route-type level 1 level 2`	Matches against a type of route. The *route-type* can be one or more of the following: • external • internal • level-1 • level-2 • local • nssa-external • type-1 • type-2
match tag *tagid* [*tagid...*] **Example:** `switch(config-route-map)# match tag 2`	Matches a route against one or more tags for filtering or redistribution.
match vlan *vlan-id* [*vlan-range*] **Example:** `switch(config-route-map)# match vlan 3, 5-10`	Matches against a VLAN.

You can configure the following optional set parameters for route maps in route-map configuration mode:

Command	Purpose

set as-path {**tag** | **prepend** {**last-as** *number* | *as-1* [*as-2...*]}}
Example:
`switch(config-route-map)# set as-path prepend 10 100 110`
	Modifies an AS-path attribute for a BGP route. You can prepend the configured *number* of last AS numbers or a string of particular AS-path values (*as-1 as-2...as-n*).

set comm-list *name* **delete** **Example:** `switch(config-route-map)# set comm-list BGPCommunity delete`	Removes communities from the community attribute of an inbound or outbound BGP route update. Use the **ip community-list** command to create the community list.

set community {**none** | **additive** | **local-AS** | **no-advertise** | **no-export** | *community-1* [community-2...]}
Example:
`switch(config-route-map)# set community local-AS`
	Sets the community attribute for a BGP route update.
	Note When you use both the **set community** and **set comm-list delete** commands in the same sequence of a route map attribute, the deletion operation is performed before the set operation.
	Note Use the **send-community** command in BGP neighbor address family configuration mode to propagate BGP community attributes to BGP peers.

Command	Purpose
set dampening *halflife reuse suppress duration* **Example:** `switch(config-route-map)# set dampening 30 1500 10000 120`	Sets the following BGP route dampening parameters: • *halflife*—The range is from 1 to 45 minutes. The default is 15. • *reuse*—The range is from is 1 to 20000 seconds. The default is 750. • *suppress*—The range is from is 1 to 20000. The default is 2000. • *duration*—The range is from is 1 to 255 minutes. The default is 60.
set extcomm-list *name* **delete** **Example:** `switch(config-route-map)# set extcomm-list BGPextCommunity delete`	Removes communities from the extended community attribute of an inbound or outbound BGP route update. Use the **ip extcommunity-list** command to create the extended community list.

set extcommunity generic {**transitive** | **nontransitive**} {**none** | **additive**] *community-1* [community-2...]}
Example:
`switch(config-route-map)# set extcommunity generic transitive 1.0:30`
	Sets the extended community attribute for a BGP route update.
	Note When you use both the **set extcommunity** and **set extcomm-list delete** commands in the same sequence of a route map attribute, the deletion operation is performed before the set operation.
	Note Use the **send-community** command in BGP neighbor address family configuration mode to propagate BGP extended community attributes to BGP peers.

set forwarding-address **Example:** `switch(config-route-map)# set forwarding-address`	Sets the forwarding address for OSPF.

set level {**backbone** | **level-1** | **level-1-2** | **level-2**}
Example:
`switch(config-route-map)# set level backbone`
	Sets what area to import routes to for IS-IS. The options for IS-IS are level-1, level-1-2, or level-2. The default is level-1.

set local-preference *value* **Example:** `switch(config-route-map)# set local-preference 4000`	Sets the BGP local preference value. The range is from 0 to 4294967295.
set metric [**+** | **-**]*bandwidth-metric* **Example:** `switch(config-route-map)# set metric +100`	Adds or subtracts from the existing metric value. The metric is in Kb/s. The range is from 0 to 4294967295.

Command	Purpose
set metric *bandwidth* [*delay reliability load mtu*] **Example:** `switch(config-route-map)# set metric 33 44 100 200 1500`	Sets the route metric values. Metrics are as follows: • *metric0*—Bandwidth in Kb/s. The range is from 0 to 4294967295. • *metric1*—Delay in 10-microsecond units. • *metric2*—Reliability. The range is from 0 to 255 (100 percent reliable). • *metric3*—Loading. The range is from 1 to 200 (100 percent loaded). • *metric4*—MTU of the path. The range is from 1 to 4294967295.

set metric-type {**external** | **internal** | **type-1** | **type-2**}
Example:
`switch(config-route-map)# set metric-type internal`
	Sets the metric type for the destination routing protocol. The options are as follows:
	external—IS-IS external metric
	internal— IGP metric as the MED for BGP
	type-1—OSPF external type 1 metric
	type-2—OSPF external type 2 metric

set origin {**egp** *as-number* | **igp** | **incomplete**}
Example:
`switch(config-route-map)# set origin incomplete`
	Sets the BGP origin attribute. The EGP *as-number* range is from 0 to 65535.

set tag *name* **Example:** `switch(config-route-map)# set tag 33`	Sets the tag value for the destination routing protocol. The *name* parameter is an unsigned integer.
set weight *count* **Example:** `switch(config-route-map)# set weight 33`	Sets the weight for the BGP route. The range is from 0 to 65535.

The **set metric-type internal** command affects an outgoing policy and an eBGP neighbor only. If you configure both the **metric** and **metric-type internal** commands in the same BGP peer outgoing policy, then Cisco NX-OS ignores the **metric-type internal** command.

Verifying the Route Policy Manager Configuration

To display the route policy manager configuration information, perform one of the following tasks:

Command	Purpose
show ip community-list [*name*]	Displays information about a community list.
show ip extcommunity-list [*name*]	Displays information about an extended community list.

Command	Purpose
show [**ip**] **prefix-list** [*name*]	Displays information about an IPv4 prefix list.
show route-map [*name*]	Displays information about a route map.

Configuration Examples for Route Policy Manager

This example shows how to use an address family to configure BGP so that any unicast and multicast routes from neighbor 209.0.2.1 are accepted if they match access list 1:

```
router bgp 64496
  address-family ipv4 unicast
    network 192.0.2.0/24
    network 209.165.201.0/27 route-map filterBGP
route-map filterBGP
 match ip next-hop prefix-list AllowPrefix
ip prefix-list AllowPrefix 10 permit 192.0.2.0 eq 24
ip prefix-list AllowPrefix 20 permit 209.165.201.0 eq 27
```

Related Topics

The following topics can give more information on Route Policy Manager:

- Chapter 1, “Configuring Basic BGP”
- Chapter 1, “Managing the Unicast RIB and FIB”

Additional References

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

- Related Documents, page 1-19
- Standards, page 1-19

Related Documents

Related Topic	Document Title
Route Policy Manager CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

Configuring Policy Based Routing

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

This chapter includes the following sections:

Information About Policy Based Routing

Policy-based routing allows you to configure a defined policy for IPv4 and IPv6 traffic flows, lessening reliance on routes derived from routing protocols. All packets received on an interface with policy-based routing enabled are passed through enhanced packet filters or *route maps*. The route maps dictate the policy, determining where to forward packets.

Route maps are composed of match and set statements that you can mark as permit or deny. You can interpret the statements as follows:

- If the packets match any route map statements, all the set statements are applied. One of these actions involves choosing the next hop.
- If the statement is marked as permit and the packets do not match any route-map statements, the packets are sent back through the normal forwarding channels and destination-based routing is performed.

For more information, see the "Route Maps" section on page 1-2.

Policy-based routing includes the following features:

- Source-based routing—Routes traffic that originates from different sets of users through different connections across the policy routers.

This section includes the following topics:

- Policy Route Maps, page 1-2
- Set Criteria for Policy-Based Routing, page 1-2

Policy Route Maps

Each entry in a route map contains a combination of match and set statements. The match statements define the criteria for whether appropriate packets meet the particular policy (that is, the conditions to be met). The set clauses explain how the packets should be routed once they have met the match criteria.

You can mark the route-map statements as permit or deny. If the statement is marked as a deny, the packets that meet the match criteria are sent back through the normal forwarding channels (destination-based routing is performed). If the statement is marked as permit and the packets meet the match criteria, all the set clauses are applied. If the statement is marked as permit and the packets do not meet the match criteria, those packets are also forwarded through the normal routing channel.

Note Policy routing is specified on the interface that receives the packets, not on the interface from which the packets are sent.

Set Criteria for Policy-Based Routing

The set criteria in a route map is evaluated in the order listed in the route map. Set criteria specific to route maps used for policy-based routing are as follows:

- List of specified IP addresses—The IP address can specify the adjacent next-hop router in the path toward the destination to which the packets should be forwarded. The first IP address associated with a connected interface that is currently up is used to route the packets.

If the packets do not meet any of the defined match criteria, the packets are routed through the normal destination-based routing process.

Licensing Requirements for Policy-Based Routing

The following table shows the licensing requirements for this feature:

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

Prerequisites for Policy-Based Routing

Policy-based routing has the following prerequisites:

- Install the correct license.
- You must enable policy-based routing (see the "Enabling the Policy-Based Routing Feature" section on page 1-3).

- Assign an IP address on the interface and bring the interface up before you apply a route map on the interface for policy-based routing.

Guidelines and Limitations for Policy-Based Routing

Policy-based routing has the following configuration guidelines and limitations:

- A policy-based routing route map can have only one match or set statement per route-map statement.
- A **match** command can refer to only one ACL in a route map used for policy-based routing.
- An ACL used in a policy-based routing route map cannot include a deny statement.
- The same route map can be shared among different interfaces for policy-based routing as long as the interfaces belong to the same virtual routing and forwarding (VRF) instance.
- Setting a tunnel interface or an IP address via a tunnel interface as a next hop in a policy-based routing policy is not supported.
- The Cisco Nexus 6000 does not support multi-sequence configuration in policy-based routing.

Default Settings

Table 1-1 lists the default settings for policy-based routing parameters.

Table 1-1 Default Policy-based Routing Parameters

Parameters	Default
Policy-based routing	Disabled

Configuring Policy-Based Routing

This section includes the following topics:

- Enabling the Policy-Based Routing Feature, page 1-3
- Configuring a Route Policy, page 1-4

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 the Policy-Based Routing Feature

You must enable the policy-based routing feature before you can configure a route policy.

SUMMARY STEPS

1. **configure terminal**
2. **feature pbr**

3. **(Optional) show feature**
4. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`feature pbr`**
	Example:
	`switch(config)# feature pbr`
		Enables the policy-based routing feature.

Step 3	**`show feature`**
	Example:
	`switch(config)# show feature`
		(Optional) Displays enabled and disabled features.

Step 4	**`copy running-config startup-config`**
	Example:
	`switch(config)# copy running-config startup-config`
		(Optional) Saves this configuration change.

Use the **no feature pbr** command to disable the policy-based routing feature and remove all associated configuration.

Command	Purpose

`no feature pbr`
Example:
`switch(config)# no feature pbr`
	Disables policy-based routing and removes all associated configuration.

Configuring a Route Policy

You can use route maps in policy-based routing to assign routing policies to the inbound interface. See the "Configuring Route Maps" section on page 1-12.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type slot/port*
3. **ip policy route-map** *map***-***name*

   or

   ipv6 policy route-map *map***-***nam*
4. **(Optional) exit**
5. **(Optional) exit**

6. **(Optional) copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

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

Step 3	**`ip policy route-map`** *`map-name`*
	Example:
	`switch(config-if)# ip policy route-map Testmap`
		Assigns a route map for IPv4 policy-based routing to the interface.

	`ipv6 policy route-map` *`map-name`*
	Example:
	`switch(config-if)# ipv6 policy route-map TestIPv6map`
		Assigns a route map for IPv6 policy-based routing to the interface.

Step 4	**`exit`**
	Example:
	`switch(config-route-map)# exit`
		(Optional) Exits route-map configuration mode.

Step 5	**`exit`**
	Example:
	`switch(config)# exit`
		(Optional) Exits global configuration mode.

Step 6	**`copy running-config startup-config`**
	Example:
	`switch# copy running-config startup-config`
		(Optional) Saves this configuration change.

This example shows how to add a route map to an interface:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# ip policy route-map Testmap
switch(config)# exit
switch(config)# copy running-config startup-config
```

You can configure the following optional match parameters for route maps in route-map configuration mode:

Command	Purpose
match ip address *acess-list-name* **Example:** `switch(config-route-map)# match ip address ACL1`	Matches an IPv4 address against an IP access control list (ACL). This command is used for policy-based routing and is ignored by route filtering or redistribution.
match ipv6 address *acess-list-name* **Example:** `switch(config-route-map)# match ipv6 address ACLv6`	Matches an IPv6 address against an IPv6 ACL. This command is used for policy-based routing and is ignored by route filtering or redistribution.

You can configure the following optional set parameters for route maps in route-map configuration mode:

Command	Purpose
set ip next-hop *address1* [*address2...*] **Example:** `switch(config-route-map)# set ip next-hop 192.0.2.1`	Sets the IPv4 next-hop address for policy-based routing. This command uses the first valid next-hop address if multiple addresses are configured.
set ipv6 next-hop *address1* [*address2...*] **Example:** `switch(config-route-map)# set ipv6 next-hop 2001:0DB8::1`	Sets the IPv6 next-hop address for policy-based routing. This command uses the first valid next-hop address if multiple addresses are configured.

set interface {**null0**}
Example:
`switch(config-route-map)# set interface null0`
	Sets the interface used for routing. Use the **null0** interface to drop packets.

Cisco Nexus 6000 routes the packet as soon as it finds a next hop and an interface.

Verifying the Policy-Based Routing Configuration

To display policy-based routing configuration information, perform one of the following tasks:

Command	Purpose
show [**ip** | **ipv6**] **policy** [*name*]	Displays information about an IPv4 or IPv6 policy.
show route-map [*name*] **pbr-statistics**	Displays policy statistics.

Use the **route-map** *map-name* **pbr-statistics** to enable policy statistics. Use the **clear route-map** *map-name* **pbr-statistics** to clear these policy statistics

Configuration Examples for Policy-Based Routing

This example shows how to configure a simple route policy on an interface:

```
feature pbr
ip access-list pbr-sample
  permit tcp host 10.1.1.1 host 192.168.2.1 eq 80
!
route-map pbr-sample
 match ip address pbr-sample
 set ip next-hop 192.168.1.1
!
route-map pbr-sample pbr-statistics

 interface ethernet 1/2
  ip policy route-map pbr-sample
```

The following output verifies this configuration:

```
switch# show route-map pbr-sample

route-map pbr-sample, permit, sequence 10
 Match clauses:
   ip address (access-lists): pbr-sample
 Set clauses:
   ip next-hop 192.168.1.1

switch# show route-map pbr-sample pbr-statistics

route-map pbr-sample, permit, sequence 10
 Policy routing matches: 84 packets
```

Related Topics

The following topics can give more information on Policy Based Routing:

- Chapter 1, “Configuring Route Policy Manager”

Additional References

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

- Related Documents, page 1-8
- Standards, page 1-8

Related Documents

Related Topic	Document Title
Policy-based routing CLI commands	*Cisco Nexus 6000 Series NX-OS Unicast Routing Command Reference, Release 6.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

CHAPTER 1

Configuring HSRP

This chapter describes how to configure the Hot Standby Router Protocol (HSRP) on the Cisco NX-OS switch.

This chapter includes the following sections:

Information About HSRP

HSRP is a first-hop redundancy protocol (FHRP) that allows a transparent failover of the first-hop IP router. HSRP provides first-hop routing redundancy for IP hosts on Ethernet networks configured with a default router IP address. You use HSRP in a group of routers for selecting an active router and a standby router. In a group of routers, the active router is the router that routes packets; the standby router is the router that takes over when the active router fails or when preset conditions are met.

Many host implementations do not support any dynamic router discovery mechanisms but can be configured with a default router. Running a dynamic router discovery mechanism on every host is not feasible for a number of reasons, including administrative overhead, processing overhead, and security issues. HSRP provides failover services to these hosts.

This section includes the following topics:

- HSRP Messages, page 1-5
- HSRP Load Sharing, page 1-6
- BFD, page 1-7
- vPC and HSRP, page 1-7
- Virtualization Support, page 1-7

HSRP Overview

When you use HSRP, you configure the HSRP virtual IP address as the host's default router (instead of the IP address of the actual router). The virtual IP address is an IPv4 or IPv6 address that is shared among a group of routers that run HSRP.

When you configure HSRP on a network segment, you provide a virtual MAC address and a virtual IP address for the HSRP group. You configure the same virtual address on each HSRP-enabled interface in the group. You also configure a unique IP address and MAC address on each interface that acts as the real address. HSRP selects one of these interfaces to be the active router. The active router receives and routes packets destined for the virtual MAC address of the group.

HSRP detects when the designated active router fails. At that point, a selected standby router assumes control of the virtual MAC and IP addresses of the HSRP group. HSRP also selects a new standby router at that time.

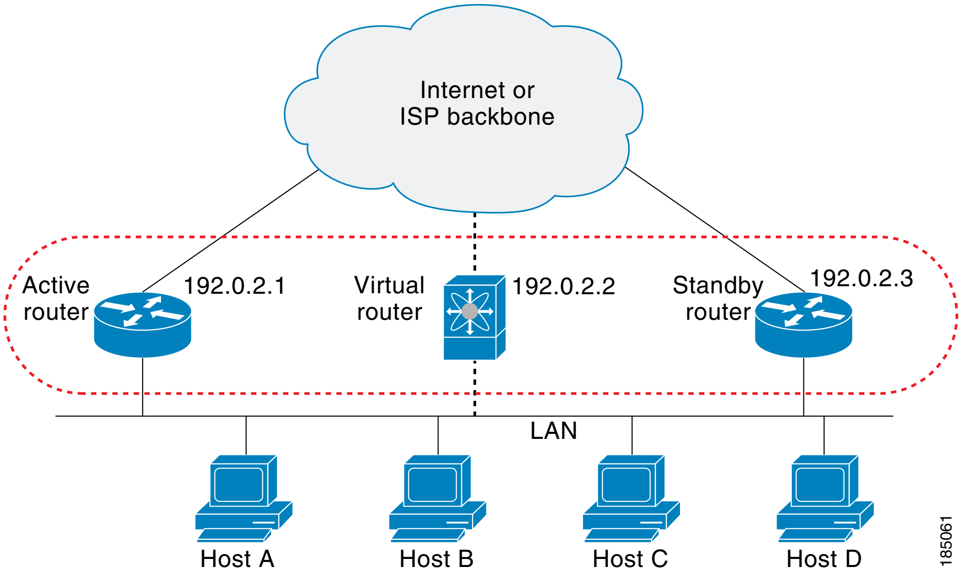
HSRP uses a priority mechanism to determine which HSRP-configured interface becomes the default active router. To configure an interface as the active router, you assign it with a priority that is higher than the priority of all the other HSRP-configured interfaces in the group. The default priority is 100, so if you configure just one interface with a higher priority, that interface becomes the default active router.

Interfaces that run HSRP send and receive multicast User Datagram Protocol (UDP)-based hello messages to detect a failure and to designate active and standby routers. When the active router fails to send a hello message within a configurable period of time, the standby router with the highest priority becomes the active router. The transition of packet forwarding functions between the active and standby router is completely transparent to all hosts on the network.

You can configure multiple HSRP groups on an interface.

Figure 1-1 shows a network configured for HSRP. By sharing a virtual MAC address and a virtual IP address, two or more interfaces can act as a single virtual router.

Figure 1-1 HSRP Topology with Two Enabled Routers



The virtual router does not physically exist but represents the common default router for interfaces that are configured to provide backup to each other. You do not need to configure the hosts on the LAN with the IP address of the active router. Instead, you configure them with the IP address (virtual IP address) of the virtual router as their default router. If the active router fails to send a hello message within the configurable period of time, the standby router takes over, responds to the virtual addresses, and becomes the active router, assuming the active router duties. From the host perspective, the virtual router remains the same.

Note Packets received on a routed port destined for the HSRP virtual IP address will terminate on the local router, regardless of whether that router is the active HSRP router or the standby HSRP router. This includes ping and Telnet traffic. Packets received on a Layer 2 (VLAN) interface destined for the HSRP virtual IP address will terminate on the active router.

HSRP for IPv4

HSRP routers communicate with each other by exchanging HSRP hello packets. These packets are sent to the destination IP multicast address 224.0.0.2 (reserved multicast address used to communicate to all routers) on UDP port 1985. The active router sources hello packets from its configured IP address and the HSRP virtual MAC address while the standby router sources hellos from its configured IP address and the interface MAC address, which may or may not be the burned-in address (BIA). The BIA is the last six bytes of the MAC address that is assigned by the manufacturer of the network interface card (NIC).

Because hosts are configured with their default router as the HSRP virtual IP address, hosts must communicate with the MAC address associated with the HSRP virtual IP address. This MAC address is a virtual MAC address, 0000.0C07.ACxy, where xy is the HSRP group number in hexadecimal based on the respective interface. For example, HSRP group 1 uses the HSRP virtual MAC address of 0000.0C07.AC01. Hosts on the adjoining LAN segment use the normal Address Resolution Protocol (ARP) process to resolve the associated MAC addresses.

HSRP version 2 uses the new IP multicast address 224.0.0.102 to send hello packets instead of the multicast address of 224.0.0.2, which is used by version 1. HSRP version 2 permits an expanded group number range of 0 to 4095 and uses a new MAC address range of 0000.0C9F.F000 to 0000.0C9F.FFFF

.HSRP for IPv6

IPv6 hosts learn of available IPv6 routers through IPv6 neighbor discovery (ND) router advertisement (RA) messages. These messages are multicast periodically, or be solicited by hosts, but the time delay for detecting when a default route is down be 30 seconds or more. HSRP for IPv6 provides a much faster switchover to an alternate default router than the IPv6 ND protocol provides, less than a second if the milliseconds timers are used. HSRP for IPv6 provides a virtual first hop for IPv6 hosts.

When you configure an IPv6 interface for HSRP, the periodic RAs for the interface link-local address stop after IPv6 ND sends a final RA with a router lifetime of zero. No restrictions occur for the interface IPv6 link-local address. Other protocols continue to receive and send packets to this address.

IPv6 ND sends periodic RAs for the HSRP virtual IPv6 link-local address when the HSRP group is active. These RAs stop after a final RA is sent with a router lifetime of 0 when the HSRP group leaves the active state. HSRP uses the virtual MAC address for active HSRP group messages only (hello, coup, and redesign).

HSRP for IPv6 uses the following parameters:

- HSRP version 2
- UDP port 2029
- Virtual MAC address range from 0005.73A0.0000 through 0005.73A0.0FFF
- Multicast link-local IP destination address of FF02::66
- Hop limit set to 255

HSRP IPv6 Addresses

An HSRP IPv6 group has a virtual MAC address that is derived from the HSRP group number and a virtual IPv6 link-local address that is derived, by default, from the HSRP virtual MAC address. The default virtual MAC address for an HSRP IPv6 group always used to form the virtual IPv6 link-local address, regardless of the actual virtual MAC address used by the group.

Table 1-1 shows the MAC and IP addresses used for IPv6 neighbor discovery packets and HSRP packets.

Table 1-1 HSRP and IPv6 ND Addresses

Packet	MAC Source Address	IPv6 Source Address	IPv6 Destination Address	Link-layer Address Option
Neighbor solicitation (NS)	Interface MAC address	Interface IPv6 address	—	Interface MAC address
Router solicitation (RS)	Interface MAC address	Interface IPv6 address	—	Interface MAC address
Neighbor advertisement (NA)	Interface MAC address	Interface IPv6 address	Virtual IPv6 address	HSRP virtual MAC address
Route advertisement (RA)	Interface MAC address	Virtual IPv6 address	—	HSRP virtual MAC address
HSRP (inactive)	Interface MAC address	Interface IPv6 address	—	—
HSRP (active)	Virtual MAC address	Interface IPv6 address	—	—

HSRP does not add IPv6 link-local addresses to the Unicast Routing Information Base (URIB). There are also no secondary virtual IP addresses for link-local addresses.

For global unicast addresses, HSRP adds the virtual IPv6 address to the URIB and IPv6 but does not register the virtual IPv6 addresses to ICMPv6. ICMPv6 redirects are not supported for HSRP IPv6 groups.

HSRP Versions

Cisco NX-OS supports HSRP version 1 by default. You can configure an interface to use HSRP version 2.

HSRP version 2 has the following enhancements to HSRP version 1:

- Expands the group number range. HSRP version 1 supports group numbers from 0 to 255. HSRP version 2 supports group numbers from 0 to 4095.
- For IPv4, uses the IPv4 multicast address 224.0.0.102 or the IPv6 multicast address FF02::66 to send hello packets instead of the multicast address of 224.0.0.2, which is used by HSRP version 1.
- Uses the MAC address range from 0000.0C9F.F000 to 0000.0C9F.FFFF for IPv4 and 0005.73A0.0000 through 0005.73A0.0FFF for IPv6 addresses. HSRP version 1 uses the MAC address range 0000.0C07.AC00 to 0000.0C07.ACFF.
- Adds support for MD5 authentication.

When you change the HSRP version, Cisco NX-OS reinitializes the group because it now has a new virtual MAC address.

HSRP version 2 has a different packet format than HSRP version 1. The packet format uses a type-length-value (TLV) format. HSRP version 2 packets received by an HSRP version 1 router are ignored.

HSRP Authentication

HSRP message digest 5 (MD5) algorithm authentication protects against HSRP-spoofing software and uses the industry-standard MD5 algorithm for improved reliability and security. HSRP includes the IPv4 or IPv6 address in the authentication TLVs .

HSRP Messages

Routers that are configured with HSRP exchange the following three types of multicast messages:

- Hello—The hello message conveys the HSRP priority and state information of the router to other HSRP routers.
- Coup—When a standby router wants to assume the function of the active router, it sends a coup message.
- Resign—A router that is the active router sends this message when it is about to shut down or when a router that has a higher priority sends a hello or coup message.

HSRP Load Sharing

HSRP allows you to configure multiple groups on an interface. You can configure two overlapping IPv4 HSRP groups to load share traffic from the connected hosts while providing the default router redundancy expected from HSRP. Figure 1-2 shows an example of a load-sharing HSRP IPv4 configuration.

Figure 1-2 HSRP Load Sharing

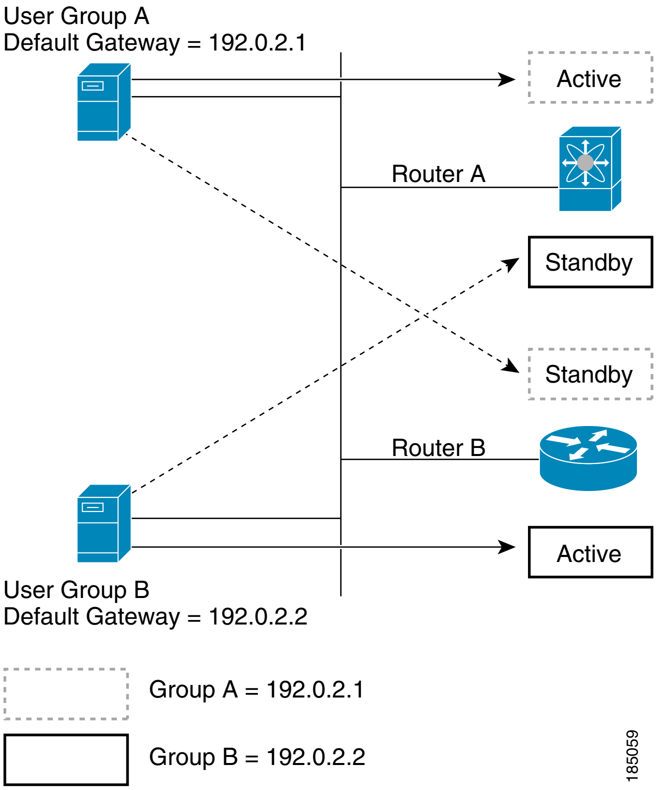


Figure 1-2 shows two routers (A and B) and two HSRP groups. Router A is the active router for group A but is the standby router for group B. Similarly, router B is the active router for group B and the standby router for group A. If both routers remain active, HSRP load balances the traffic from the hosts across both routers. If either router fails, the remaining router continues to process traffic for both hosts

Note HSRP for IPv6 load balances by default. If there are two HSRP IPv6 groups on the subnet, hosts learn of both from their router advertisements and choose to use one so that the load is shared between the advertised routers.

BFD

HSRP supports Bidirectional forwarding detection (BFD). BFD is a detection protocol that provides fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules. See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x* for more information.

vPC and HSRP

HSRP interoperates with virtual port channels (vPCs). vPCs allow links that are physically connected to two different Cisco Nexus 6000 switches to appear as a single port channel by a third switch. See the *Cisco Nexus 6000 Series NX-OS Layer 2 Switching Configuration Guide, Release 6.0*, for more information on vPCs.

vPC forwards traffic through both the active HSRP router and the standby HSRP router. You can configure a threshold on the priority of the standby HSRP router to determine when traffic should fail over to the vPC trunk. See the "Configuring the HSRP Priority" section on page 1-16.

Note You should configure HSRP on the primary vPC peer switch as active and HSRP on the vPC secondary switch as standby.

vPC Peer Gateway and HSRP

Some third-party devices can ignore the HSRP virtual MAC address and instead use the source MAC address of an HSRP router. in a vPC environment, the packets using this source MAC address may be sent across the vPC peer link, causing a potential dropped packet. Configure the vPC peer gateway to enable the HSRP routers to directly handle packets sent to the local vPC peer MAC address and the remote vPC peer MAC address, as well as the HSRP virtual MAC address. See the *Cisco Nexus 6000 Series NX-OS Layer 2 Switching Configuration Guide, Release 6.0*, for more information on the vPC peer gateway.

Note For mixed-chassis configurations where the vPC peer link is configured on an F-series module, configure the vPC peer gateway exclude option to exclude the Layer 3 backup route that traverses the vPC peer link. See the *Cisco Nexus 6000 Series NX-OS Layer 2 Switching Configuration Guide, Release 6.0*, for more information on the vPC peer gateway exclude option.

Virtualization Support

HSRP supports Virtual Routing and Forwarding instances (VRFs).

If you change the VRF membership of an interface, Cisco NX-OS removes all Layer 3 configuration, including HSRP.

Licensing Requirements for HSRP

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	HSRP requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*. **Note** Make sure the Layer 3 Hardware and LAN Base Services licenses that are included with the hardware are installed on the switch to enable Layer 3 interfaces.

Prerequisites for HSRP

HSRP has the following prerequisites:

- You must enable the HSRP feature in a switch before you can configure and enable any HSRP groups.

Guidelines and Limitations

HSRP has the following configuration guidelines and limitations:

- The minimum hello timer value is 250 milliseconds.
- The minimum hold timer value is 750 milliseconds.
- You must configure an IP address for the interface that you configure HSRP on and enable that interface before HSRP becomes active.
- You must configure HSRP version 2 when you configure an IPv6 interface for HSRP.
- For IPv4, the virtual IP address must be in the same subnet as the interface IP address.
- We recommend that you do not configure more than one first-hop redundancy protocol on the same interface.HSRP version 2 does not interoperate with HSRP version 1. An interface cannot operate both version 1 and version 2 because both versions are mutually exclusive. However, the different versions can be run on different physical interfaces of the same router.
- You cannot change from version 2 to version 1 if you have configured groups above the group number range allowed for version 1 (0 to 255).
- Cisco NX-OS removes all Layer 3 configuration on an interface when you change the interface VRF membership, port channel membership, or when you change the port mode to Layer 2.
- If you configure virtual MAC addresses with a virtual port channel (vPC), you must configure the same virtual MAC address on both vPC peers.
- You cannot use the HSRP MAC address burned-in option on a VLAN interface that is a vPC member.
- If the Layer 3 license is not installed on your Cisco Nexus 6000 device, HSRP can still be configured but will not function and a non-disruptive ISSU is not possible.
- All Layer 3 configuration must be removed from the Cisco Nexus 6000 device before clearing the Layer 3 license, including OSPF, PIM, and **no switchport** configurations. HSPR does not need to be removed before clearing the Layer 3 license but it is recommended that it be unconfigured first.
- If you have not configured authentication, the **show hsrp** command displays the following string: Authentication text "cisco".

- This is the default behavior of HSRP as defined in RFC 2281: If no authentication data is configured, the RECOMMENDED default value is 0x63 0x69 0x73 0x63 0x6F 0x00 0x00 0x00.

Default Settings

Table 1-2 lists the default settings for HSRP parameters.

Table 1-2 Default HSRP Parameters

Parameters	Default
HSRP	Disabled
Authentication	Enabled as text for version 1, with cisco as the password
HSRP version	Version 1
Preemption	disabled
Priority	100
virtual MAC address	Derived from HSRP group number

Configuring HSRP

This section includes the following topics:

- Enabling the HSRP Feature, page 1-10
- Configuring the HSRP Version, page 1-10
- Configuring an HSRP Group for IPv4, page 1-10
- Configuring an HSRP Group for IPv6, page 1-12
- Configuring the HSRP Virtual MAC Address, page 1-14
- Authenticating HSRP, page 1-15switch(config-if-hsrp)# copy running-config startup-config, page 1-16
- Configuring the HSRP Priority, page 1-16
- Customizing HSRP, page 1-17

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 the HSRP Feature

You must globally enable the HSRP feature before you can configure and enable any HSRP groups.

To enable the HSRP feature, use the following command in global configuration mode:

DETAILED STEPS

Command	Purpose
feature hsrp **Example:** `switch(config)# feature hsrp`	Enables HSRP.

To disable the HSRP feature and remove all associated configuration, use the following command in global configuration mode:

Configuring the HSRP Version

Command	Purpose
no feature hsrp **Example:** `switch(config)# no feature hsrp`	Disables HSRP for all groups.

You can configure the HSRP version. If you change the version for existing groups, Cisco NX-OS reinitializes HSRP for those groups because the virtual MAC address changes. The HSRP version applies to all groups on the interface.

Note IPv6 HSRP groups must be configured as HSRP version 2.

To configure the HSRP version, use the following command in interface configuration mode:

Command	Purpose

hsrp version {**1** | **2**}
Example:
`switch(config-if)# hsrp version 2`
	Configures the HSRP version. Version 1 is the default.

Configuring an HSRP Group for IPv4

You can configure an HSRP group on an IPv4 interface and configure the virtual IP address and virtual MAC address for the HSRP group.

BEFORE YOU BEGIN

Ensure that you have enabled the HSRP feature (see the “Enabling the HSRP Feature” section on page 1-10).

Cisco NX-OS enables an HSRP group once you configure the virtual IP address on any member interface in the group. You should configure HSRP attributes such as authentication, timers, and priority before you enable the HSRP group.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*
3. **no switchport**
4. **ip** *ip-address/length*
5. **hsrp** *group-number* [**ipv4**]
6. **ip** [*ip-address* [**secondary**]]
7. **exit**
8. **no shutdown**
9. (Optional) **show hsrp [group** *group-number*] [**ipv4**]
10. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

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

Step 3	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	**`ip`** *`ip-address/length`*
	Example:
	`switch(config-if)# ip 192.0.2.2/8`
		Configures the IPv4 address of the interface.

Step 5	**`hsrp`** *`group-number`* `[`**`ipv4`**`]`
	Example:
	`switch(config-if)# hsrp 2`
	`switch(config-if-hsrp)#`
		Creates an HSRP group and enters hsrp configuration mode. The range for HSRP version 1 is from 0 to 255. The range is for HSRP version 2 is from 0 to 4095. The default value is 0.

Step 6	**`ip`** `[`*`ip-address`* `[`**`secondary`**`]]`
	Example:
	`switch(config-if-hsrp)# ip 192.0.2.1`
		Configures the virtual IP address for the HSRP group and enables the group. This address should be in the same subnet as the IPv4 address of the interface.

	Command	Purpose
Step 7	**exit** **Example**: switch(config-if-hsrp)# exit	Exits HSRP configuration mode.
Step 8	**no shutdown** **Example**: switch(config-if)# no shutdown	Enables the interface.
Step 9	**show hsrp** [**group** *group-number*] [**ipv4**] **Example**: switch(config-if)# show hsrp group 2	(Optional) Displays HSRP information.
Step 10	**copy running-config startup-config** **Example:** switch(config-if)# copy running-config startup-config	(Optional) Saves this configuration change.

Note You should use the **no shutdown** command to enable the interface after you finish the configuration.

This example shows how to configure an HSRP group on Ethernet 1/2:

```
switch# configure terminal
switch(config)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# ip 192.0.2.2/8
switch(config-if)# hsrp 2
switch(config-if-hsrp)# ip 192.0.2.1
switch(config-if-hsrp)# exit
switch(config-if)# no shutdown
switch(config-if)# copy running-config startup-config
```

Configuring an HSRP Group for IPv6

You can configure an HSRP group on an IPv6 interface and configure the virtual MAC address for the HSRP group.

When you configure an HSRP group for IPv6, HSRP generates a link-local address from the link-local prefix. HSRP also generates a modified EUI-64 format interface identifier in which the EUI-64 interface identifier is created from the relevant HSRP virtual MAC address.

There are no HSRP IPv6 secondary addresses.

BEFORE YOU BEGIN

Ensure that you have enabled the HSRP feature (see the “Enabling the HSRP Feature” section on page 1-10).

Ensure that you have enabled HSRP version 2 on the interface that you want to configure an IPv6 HSRP group on.

Ensure that you have configured HSRP attributes such as authentication, timers, and priority before you enable the HSRP group.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *type number*
3. **ipv6** *ipv6-address/length*
4. **hsrp version 2**
5. **hsrp** *group-number* **ipv6**
6. **ip** *ipv6-address* **[secondary]**
7. **ip autoconfig**
8. **no shutdown**
9. **show hsrp [group** *group-number*] [**ipv6**]
10. **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`type number`*
	Example:
	`switch(config)# interface ethernet 3/2`
	`switch(config-if)#`
		Enters interface configuration mode.

Step 3	**`ipv6`** *`ipv6-address/length`*
	Example:
	`switch(config-if)# ipv6 2001:0DB8:0001:0001:/64`
		Configures the IPv6 address of the interface.

Step 4	**`hsrp version 2`**
	Example:
	`switch(config-if-hsrp)# hsrp version 2`
		Configures this group for HSRP version 2.

Step 5	**`hsrp`** *`group-number`* **`ipv6`**
	Example:
	`switch(config-if)# hsrp 10 ipv6`
	`switch(config-if-hsrp)#`
		Creates an IPv6 HSRP group and enters hsrp configuration mode. The range for HSRP version 2 is from 0 to 4095. The default value is 0.

Step 6	**`ip`** [*`ipv6-address`* [**`secondary`**]] **Example:** `switch(config-if-hsrp)# ip 2001:DB8::1`	Configures the virtual IPv6 address for the HSRP group and enables the group.

Step 7	**`ip autoconfig`**
	Example:
	`switch(config-if-hsrp)# ip autoconfig`
		Autoconfigures the virtual IPv6 address for the HSRP group from the calculated link-local virtual IPv6 address and enables the group.

	Command	Purpose
Step 8	**no shutdown** **Example:** switch(config-if-hsrp)# no shutdown	Enables the interface.
Step 9	**show hsrp** [**group** *group-number*] [**ipv6**] **Example:** switch(config-if-hsrp)# show hsrp group 10	(Optional) Displays HSRP information.
Step 10	**copy running-config startup-config** **Example:** switch(config-if-hsrp)# copy running-config startup-config	(Optional) Saves this configuration change.

Note You should use the **no shutdown** command to enable the interface after you finish the configuration.

The following example shows how to configure an IPv6 HSRP group on Ethernet 3/2:

```
switch# configure terminal
switch(config)# interface ethernet 3/2
switch(config-if)# ip 12001:0DB8:0001:0001:/64
switch(config-if)# hsrp 2 ipv6
switch(config-if-hsrp)# exit
switch(config-if)# no shutdown
switch(config-if)# copy running-config startup-config
```

Configuring the HSRP Virtual MAC Address

You can override the default virtual MAC address that HSRP derives from the configured group number.

Note You must configure the same virtual MAC address on both vPC peers of a vPC link.

To manually configure the virtual MAC address for an HSRP group, use the following command in hsrp configuration mode:

Command	Purpose
mac-address *string* **Example:** switch(config-if-hsrp)# mac-address 5000.1000.1060	Configures the virtual MAC address for an HSRP group. The string uses the standard MAC address format (xxxx.xxxx.xxxx).

To configure HSRP to use the burned-in MAC address of the interface for the virtual MAC address, use the following command in interface configuration mode:

Command	Purpose
hsrp use-bia [**scope interface**] **Example:** `switch(config-if)# hsrp use-bia`	Configures HSRP to use the burned-in MAC address of the interface for the HSRP virtual MAC address. You can optionally configure HSRP to use the burned-in MAC address for all groups on this interface by using the **scope interface** keyword.

Authenticating HSRP

You can configure HSRP to authenticate the protocol using cleartext or MD5 digest authentication. MD5 authentication uses a key chain (see the *Cisco Nexus 6000 Series NX-OS Security Configuration Guide, Release 6.0*).

BEFORE YOU BEGIN

Ensure that you have enabled the HSRP feature (see the "Enabling the HSRP Feature" section on page 1-10).

You must configure the same authentication and keys on all members of the HSRP group.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **hsrp** *group-number* [**ipv4** | **ipv6**]
5. **authentication text** *string*
   or
   authentication md5 {**key-chain** *key-chain* | **key-string** {**0** | **7**} *text* [**timeout** *seconds*]}
6. (Optional) **show hsrp** [**group** *group-number*]
7. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface** *interface-type slot/port*
	Example:
	`switch(config)# interface ethernet 1/2`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

	Command	Purpose
Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.

Step 4	**hsrp** *group-number* [**ipv4** | **ipv6**]
	Example:
	`switch(config-if)# hsrp 2`
	`switch(config-if-hsrp)#`
		Creates an HSRP group and enters HSRP configuration mode.

Step 5	**authentication text** *string* **Example:** `switch(config-if-hsrp)# authentication text mypassword`	Configures cleartext authentication for HSRP on this interface.

	authentication md5 {**key-chain** *key-chain* | **key-string** {**0** | **7**} *text* [**timeout** *seconds*]}
	Example:
	`switch(config-if-hsrp)# authentication md5 key-chain hsrp-keys`
		Configures MD5 authentication for HSRP on this interface. You can use a key chain or key string. If you use a key string, you can optionally set the timeout for when HSRP will only accept a new key. The range is from 0 to 32767 seconds.

Step 6	**show hsrp** [**group** *group-number*] **Example:** `switch(config-if-hsrp)# show hsrp group 2`	(Optional) Displays HSRP information.
Step 7	**copy running-config startup-config** **Example:** `switch(config-if-hsrp)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure MD5 authentication for HSRP on Ethernet 1/2 after creating the key chain:

```
switch# configure terminal
switch(config)# key chain hsrp-keys
switch(config-keychain)# key 0
switch(config-keychain-key)# key-string 7 zqdest
switch(config-keychain-key) accept-lifetime 00:00:00 Jun 01 2008 23:59:59 Sep 12 2008
switch(config-keychain-key) send-lifetime 00:00:00 Jun 01 2008 23:59:59 Aug 12 2008
switch(config-keychain-key) key 1
switch(config-keychain-key) key-string 7 uaeqdyito
switch(config-keychain-key) accept-lifetime 00:00:00 Aug 12 2008 23:59:59 Dec 12 2008
switch(config-keychain-key) send-lifetime 00:00:00 Sep 12 2008 23:59:59 Nov 12 2008
switch(config-keychain-key)# interface ethernet 1/2
switch(config-if)# no switchport
switch(config-if)# hsrp 2
switch(config-if-hsrp)# authenticate md5 key-chain hsrp-keys
switch(config-if-hsrp)# copy running-config startup-config
```

Configuring the HSRP Priority

You can configure the HSRP priority on an interface. HSRP uses the priority to determine which HSRP group member acts as the active router. If you configure HSRP on a vPC-enabled interface, you can optionally configure the upper and lower threshold values to control when to fail over to the vPC trunk

If the standby router priority falls below the lower threshold, HSRP sends all standby router traffic across the vPC trunk to forward through the active HSRP router. HSRP maintains this scenario until the standby HSRP router priority increases above the upper threshold.

For IPv6 HSRP groups, if all group members have the same priority, HSRP selects the active router based on the IPv6 link-local address.

To configure the HSRP priority, use the following command in interface configuration mode:

Command	Purpose
priority *level* [**forwarding-threshold lower** *lower-value* **upper** *upper-value*] **Example:** `switch(config-if-hsrp)# priority 60 forwarding-threshold lower 40 upper 50`	Sets the priority level used to select the active router in an HSRP group. The *level* range is from 0 to 255. The default is 100. Optionally, sets the upper and lower threshold values used by vPC to determine when to fail over to the vPC trunk. The *lower-value* range is from 1 to 255. The default is 1. The *upper-value* range is from 1 to 255. The default is 255.

Customizing HSRP

You can optionally customize the behavior of HSRP. Be aware that as soon as you enable an HSRP group by configuring a virtual IP address, that group is now operational. If you first enable an HSRP group before customizing HSRP, the router could take control over the group and become the active router before you finish customizing the feature. If you plan to customize HSRP, you should do so before you enable the HSRP group.

Command	Purpose
name *string* **Example:** `switch(config-if-hsrp)# name HSRP-1`	Specifies the IP redundancy name for an HSRP group. The *string* is from 1 to 255 characters. The default string has the following format: hsrp-<interface-short-name>-<group-id>. For example, hsrp-Eth2/1-1.

Command	Purpose
preempt [**delay** [**minimum** *seconds*] [**reload** *seconds*] [**sync** *seconds*]] **Example:** `switch(config-if-hsrp)# preempt delay minimum 60`	Configures the router to take over as an active router for an HSRP group if it has a higher priority than the current active router. This command is disabled by default. The range is from 0 to 3600 seconds.
timers [**msec**] *hellotime* [**msec**] *holdtime* **Example:** `switch(config-if-hsrp)# timers 5 18`	Configures the hello and hold time for this HSRP member as follows: • *hellotime*—The interval between successive hello packets sent. The range is from 1 to 254 seconds. • *holdtime*—The interval before the information in the hello packet is considered invalid. The range is from 3 to 255. The optional **msec** keyword specifies that the argument is expressed in milliseconds, instead of the default seconds. The timer ranges for milliseconds are as follows: • *hellotime*—The interval between successive hello packets sent. The range is from 255 to 999 milliseconds. • *holdtime*—The interval before the information in the hello packet is considered invalid. The range is from 750 to 3000 milliseconds.

To customize HSRP, use the following commands in interface configuration mode:

Command or Action	Purpose
hsrp delay minimum *seconds* **Example:** `switch(config-if)# hsrp delay minimum 30`	Specifies the minimum amount of time that HSRP waits after a group is enabled before participating in the group. The range is from 0 to 10000 seconds. The default is 0.
hsrp delay reload *seconds* **Example:** `switch(config-if)# hsrp delay reload 30`	Specifies the minimum amount of time that HSRP waits after reload before participating in the group. The range is from 0 to 10000 seconds. The default is 0.

Verifying the HSRP Configuration

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

Command	Purpose
show hsrp [**group** *group-number*]	Displays the HSRP status for all groups or one group.
show hsrp delay [**interface** *interface-type slot/port*]	Displays the HSRP delay value for all interfaces or one interface. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
show hsrp [**interface** *interface-type slot/port*]	Displays the HSRP status for an interface. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
show hsrp [**group** *group-number*] [**interface** *interface-type slot/port*] [**active**] [**all**] [**init**] [**learn**] [**listen**] [**speak**] [**standby**]	Displays the HSRP status for a group or interface for virtual forwarders in the active, init, learn, listen, or standby state. Use the **all** keyword to see all states, including disabled. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
show hsrp [**group** *group-number*] [**interface** *interface-type slot/port*] **active**] [**all**] [**init**] [**learn**] [**listen**] [**speak**] [**standby**] **brief**	Displays a brief summary of the HSRP status for a group or interface for virtual forwarders in the active, init, learn, listen, or standby state. Use the **all** keyword to see all states, including disabled. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Configuration Examples for HSRP

This example shows how to enable HSRP on an interface with MD5 authentication and interface tracking:

```
key chain hsrp-keys
 key 0
   key-string 7 zqdest
   accept-lifetime 00:00:00 Jun 01 2008 23:59:59 Sep 12 2008
   send-lifetime 00:00:00 Jun 01 2008 23:59:59 Aug 12 2008
  key 1
   key-string 7 uaeqdyito
   accept-lifetime 00:00:00 Aug 12 2008 23:59:59 Dec 12 2008
   send-lifetime 00:00:00 Sep 12 2008 23:59:59 Nov 12 2008
```

```
feature hsrp
track 2 interface ethernet 2/2 ip
interface ethernet 1/2
 no switchport
 ip address 192.0.2.2/8
 hsrp 1
  authenticate md5 key-chain hsrp-keys
  priority 90
  track 2 decrement 20
  ip-address 192.0.2.10
 no shutdown
```

Additional References

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

- Related Documents, page 1-20
- MIBs, page 1-20

Related Documents

Related Topic	Document Title
Configuring the Virtual Router Redundancy Protocol	Chapter 1, "Configuring VRRP"
HSRP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

MIBs

MIBs	MIBs Link
CISCO-HSRP-MIB	To locate and download MIBs, go to the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Configuring VRRP

This chapter describes how to configure the Virtual Router Redundancy Protocol (VRRP) on a switch

This chapter includes the following sections:

Information About VRRP

VRRP allows for transparent failover at the first-hop IP router, by configuring a group of routers to share a virtual IP address. VRRP selects a master router in that group to handle all packets for the virtual IP address. The remaining routers are in standby and take over if the master router fails.

This section includes the following topics:

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 will respond to the ARP request with its own MAC address.
- Routing protocol—The client listens to dynamic routing protocol updates (for example, from Routing Information Protocol [RIP]) and forms its own routing table.
- ICMP Router Discovery Protocol (IRDP) client—The client runs an Internet Control Message Protocol (ICMP) router discovery client.

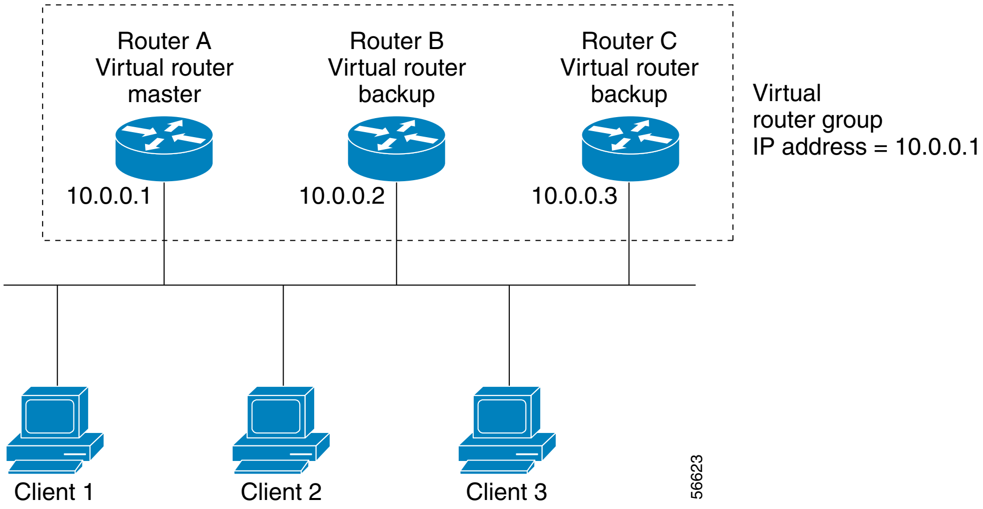
The disadvantage to dynamic discovery protocols is that they incur some configuration and processing overhead on the LAN client. Also, in the event of a router failure, the process of switching to another router can be slow.

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.

Figure 1-1 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 1-1 Basic VRRP Topology



Because the virtual IP address uses the IP address of the physical Ethernet interface of Router A, Router A is the master (also known as the IP address owner*).* As the master, Router A owns the virtual IP address of the VRRP group router 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 master fails, the backup router with the highest priority becomes the master and takes over the virtual IP address to provide uninterrupted service for the LAN hosts. When router A recovers, it becomes the router master again. For more information, see the "VRRP Router Priority and Preemption" section.

Note Packets received on a routed port destined for the VRRP virtual IP address will terminate on the local router, regardless of whether that router is the master VRRP router or a backup VRRP router. This includes ping and telnet traffic. Packets received on a Layer 2 (VLAN) interface destined for the VRRP virtual IP address will terminate on the master 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 up to 255 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 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 master 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 master for the group by altering VRRP priorities based on interface states.

Multiple VRRP Groups

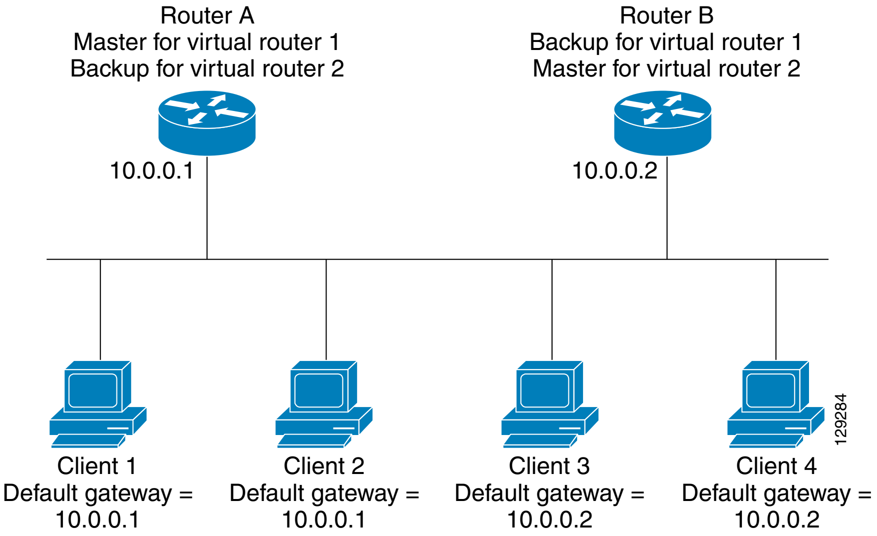
You can configure up to 255 VRRP groups on a physical interface. The actual number of VRRP groups that a router interface can support depends on the following factors:

- 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 master for one VRRP group and as a backup for one or more other VRRP groups.

Figure 1-2 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 1-2 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 master. 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 master. 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 master router fails.

If a VRRP router owns the virtual IP address and the IP address of the physical interface, this router functions as the master. The priority of the master is 255.

Priority also determines if a VRRP router functions as a backup router and the order of ascendancy to becoming a master if the master fails.

For example, if router A, the master 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 master 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 master.

VRRP uses preemption to determine what happens after a VRRP backup router becomes the master. With preemption enabled by default, VRRP will switch to a backup if that backup comes online with a priority higher than the new master. For example, if Router A is the master 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 master, even though Router B has not failed.

If you disable preemption, VRRP will only switch if the original master recovers or the new master fails.

BFD

VRRP supports Bidirectional forwarding detection (BFD). BFD is a detection protocol that provides fast forwarding-path failure detection times. BFD provides subsecond failure detection between two adjacent devices and can be less CPU-intensive than protocol hello messages because some of the BFD load can be distributed onto the data plane on supported modules. See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x* for more information.

vPC and VRRP

VRRP interoperates with virtual port channels (vPCs). vPCs allow links that are physically connected to two different Cisco Nexus 6000 switches to appear as a single port channel by a third switch. See the *Cisco Nexus 6000 Series NX-OS Layer 2 Switching Configuration Guide, Release 6.0*, for more information on vPCs.

A vPC forwards traffic through both the master VRRP router as well as the backup VRRP router. You can configure a threshold on the priority of the backup VRRP router to determine when traffic should failover to the vPC trunk. See the "Configuring VRRP Priority" section on page 1-9.

Note You should configure VRRP on the primary vPC peer switch as active and VRRP on the vPC secondary switch as standby.

VRRP Advertisements

The VRRP master sends VRRP advertisements to other VRRP routers in the same group. The advertisements communicate the priority and state of the master. 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 mechanisms:

- 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 two 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 Chapter 1, "Configuring Object Tracking" 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 may 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 master for the VRRP group. See the "Configuring VRRP Interface State Tracking" section on page 1-15 for more information.

Note VRRP does not support Layer 2 interface tracking.

Virtualization Support

VRRP supports Virtual Routing and Forwarding instances (VRFs). By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF.

If you change the VRF membership of an interface, Cisco NX-OS removes all Layer 3 configuration, including VRRP.

For more information, see Chapter 1, "Configuring Layer 3 Virtualization."

Licensing Requirements for VRRP

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	VRRP requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*. **Note** Make sure the LAN Base Services license is installed on the switch to enable Layer 3 interfaces.

Guidelines and Limitations

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 switches 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 that you configure VRRP on 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, 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.
- If the Layer 3 license is not installed on your Cisco Nexus 6000 device, VRRP can still be configured but will not function and a non-disruptive ISSU is not possible.
- All Layer 3 configuration must be removed from the Cisco Nexus 6000 device before clearing the Layer 3 license, including OSPF, PIM, and **no switchport** configurations. VRRP does not need to be removed before clearing the Layer 3 license but it is recommended that it be unconfigured first.

Default Settings

Table 1-1 lists the default settings for VRRP parameters.

Table 1-1 Default VRRP Parameters

Parameters	Default
advertisement interval	1 seconds
authentication	no authentication
preemption	enabled
priority	100
VRRP feature	disabled

Configuring VRRP

This section includes the following topics:

- Enabling the VRRP Feature, page 1-8
- Configuring VRRP Groups, page 1-8
- Configuring VRRP Priority, page 1-9
- Configuring VRRP Authentication, page 1-11
- Configuring Time Intervals for Advertisement Packets, page 1-13
- Disabling Preemption, page 1-14
- Configuring VRRP Interface State Tracking, page 1-15

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 the VRRP Feature

You must globally enable the VRRP feature before you can configure and enable any VRRP groups.

To enable the VRRP feature, use the following command in global configuration mode:

Command	Purpose
feature vrrp **Example:** `switch(config)# feature vrrp`	Enables VRRP.

To disable the VRRP feature and remove all associated configuration, use the following command in global configuration mode:

Command	Purpose
no feature vrrp **Example:** `switch(config)# no feature vrrp`	Disables the VRRP feature.

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 master VRRP router drops the packets addressed directly to the virtual IP address because the VRRP master is only intended as a next-hop router to forward packets. Some applications require that Cisco NX-OS accept packets addressed to the virtual router IP. Use the secondary option to the virtual IP address to accept these packets when the local router is the VRRP master.

Once you have configured the VRRP group, you must explicitly enable the group before it becomes active.

BEFORE YOU BEGIN

Ensure that you configure an IP address on the interface (see the "Configuring IPv4 Addressing" section on page 1-8.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrrp** *number*
5. **address** *ip-address* [**secondary**]
6. **no shutdown**
7. (Optional) **show vrrp**
8. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**interface** *interface-type slot/port* **Example:** switch(config)# switch(config-if)# interface ethernet 2/1	Enters interface configuration mode. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 3	**no switchport** **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 4	**vrrp** *number* **Example:** switch(config-if)# vrrp 250 switch(config-if-vrrp)#	Creates a virtual router group. The range is from 1 to 255.
Step 5	**address** *ip-address* [**secondary**] **Example:** 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. 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 6	**no shutdown** **Example**: switch(config-if-vrrp)# no shutdown switch(config-if-vrrp)#	Enables the VRRP group. Disabled by default.
Step 7	**show vrrp** **Example:** switch(config-if-vrrp)# show vrrp	(Optional) Displays VRRP information.
Step 8	**copy running-config startup-config** **Example:** switch(config-if-vrrp)# copy running-config startup-config	(Optional) Saves this configuration change.

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 switches whose interface IP address is the same as the primary virtual IP address (the master), the default value is 255.

If you configure VRRP on a vPC-enabled interface, you can optionally configure the upper and lower threshold values to control when to fail over to the vPC trunk If the backup router priority falls below the lower threshold, VRRP sends all backup router traffic across the vPC trunk to forward through the master VRRP router. VRRP maintains this scenario until the backup VRRP router priority increases above the upper threshold.

BEFORE YOU BEGIN

Ensure that you have enabled the VRRP feature (see the "Configuring VRRP" section on page 1-7).

Ensure that you have configured an IP address on the interface (see the "Configuring IPv4 Addressing" section on page 1-8.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrrp** *number*
5. **shutdown**
6. **priority** *level* [**forwarding-threshold lower** *lower-value* **upper** *upper-value*]
7. **no shutdown**
8. (Optional) **show vrrp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`interface-type slot/port`*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**`no switchport`**
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

Step 4	**`vrrp`** *`number`*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group.

	Command	Purpose

Step 5	`shutdown`
	Example:
	`switch(config-if-vrrp)# shutdown`
	`switch(config-if-vrrp)#`
		Disables the VRRP group. Disabled by default.

Step 6	**priority** *level* [**forwarding-threshold lower** *lower-value* **upper** *upper-value*] **Example:** `switch(config-if-vrrp)# priority 60 forwarding-threshold lower 40 upper 50`	Sets the priority level used to select the active router in an VRRP group. The *level* range is from 1 to 254. The default is 100 for backups and 255 for a master that has an interface IP address equal to the virtual IP address. Optionally, sets the upper and lower threshold values used by vPC to determine when to fail over to the vPC trunk. The *lower-value* range is from 1 to 255. The default is 1. The *upper-value* range is from 1 to 255. The default is 255.

Step 7	**no shutdown**
	Example:
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	**show vrrp** *Example:* `switch(config-if-vrrp)# show vrrp`	(Optional) Displays a summary of VRRP information.
Step 9	**copy running-config startup-config** **Example:** `switch(config-if-vrrp)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring VRRP Authentication

You can configure simple text authentication for a VRRP group.

BEFORE YOU BEGIN

Ensure that the authentication configuration is identical for all VRRP switches in the network.

Ensure that you have enabled the VRRP feature (see the “Configuring VRRP” section on page 1-7).

Ensure that you have configured an IP address on the interface (see the “Configuring IPv4 Addressing” section on page 1-8.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrrp** *number*
5. **shutdown**

6. **authentication text** *password*
7. **no shutdown**
8. (Optional) **show vrrp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**interface** *interface-type slot/port* **Example:** switch(config)# interface ethernet 2/1 switch(config-if)#	Enters interface configuration mode. **Note** If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.
Step 3	**no switchport** **Example:** switch(config-if)# no switchport	Configures the interface as a Layer 3 routed interface.
Step 4	**vrrp** *number* **Example:** switch(config-if)# vrrp 250 switch(config-if-vrrp)#	Creates a virtual router group.
Step 5	**shutdown** **Example**: switch(config-if-vrrp)# shutdown switch(config-if-vrrp)#	Disables the VRRP group. Disabled by default.
Step 6	**authentication text** *password* **Example:** switch(config-if-vrrp)# **authentication md5 prd555oln47espn0 spi 0x0**	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 7	**no shutdown** **Example**: switch(config-if-vrrp)# no shutdown switch(config-if-vrrp)#	Enables the VRRP group. Disabled by default.
Step 8	**show vrrp** **Example:** switch(config-if-vrrp)# show vrrp	(Optional) Displays a summary of VRRP information.
Step 9	**copy running-config startup-config** **Example:** switch(config-if-vrrp)# copy running-config startup-config	(Optional) Saves this configuration change.

Configuring Time Intervals for Advertisement Packets

You can configure the time intervals for advertisement packets.

BEFORE YOU BEGIN

Ensure that you have enabled the VRRP feature (see the "Configuring VRRP" section on page 1-7).

Ensure that you have configured an IP address on the interface (see the "Configuring IPv4 Addressing" section on page 1-8.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrrp** *number*
5. **shutdown**
6. **advertisement-interval** *seconds*
7. **no shutdown**
8. (Optional) **show vrrp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface** *interface-type slot/port*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.

Step 4	**vrrp** ***number***
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group.

	Command	Purpose
Step 5	**shutdown** **Example**: switch(config-if-vrrp)# shutdown switch(config-if-vrrp)#	Disables the VRRP group. Disabled by default.
Step 6	**advertisement-interval** *seconds* **Example:** switch(config-if-vrrp)# advertisement-interval 15	Sets the interval time in seconds between sending advertisement frames. The range is from 1 to 254. The default is 1 second.
Step 7	**no shutdown** **Example**: switch(config-if-vrrp)# no shutdown switch(config-if-vrrp)#	Enables the VRRP group. Disabled by default.
Step 8	**show vrrp** **Example:** switch(config-if-vrrp)# show vrrp	(Optional) Displays a summary of VRRP information.
Step 9	**copy running-config startup-config** **Example:** switch(config-if-vrrp)# copy running-config startup-config	(Optional) Saves this configuration change.

Disabling Preemption

You can disable preemption for a VRRP group member. If you disable preemption, a higher-priority backup router will not take over for a lower-priority master router. Preemption is enabled by default.

BEFORE YOU BEGIN

Ensure that you have enabled the VRRP feature (see the "Configuring VRRP" section on page 1-7).

Ensure that you have configured an IP address on the interface (see the "Configuring IPv4 Addressing" section on page 1-8.

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrrp** *number*
5. **shutdown**
6. **no preempt**
7. **no shutdown**
8. (Optional) **show vrrp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**interface** *interface-type slot/port*
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**no switchport** **Example:** `switch(config-if)# no switchport`	Configures the interface as a Layer 3 routed interface.

Step 4	**vrrp** *number*
	Example:
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group.

Step 5	**no shutdown** **Example**: `switch(config-if-vrrp)# no shutdown`	Enables the VRRP group. Disabled by default.
Step 6	**no preempt** **Example:** `switch(config-if-vrrp)# no preempt`	Disables the preempt option and allows the master to remain when a higher-priority backup appears.
Step 7	**no shutdown** **Example**: `switch(config-if-vrrp)# no shutdown`	Enables the VRRP group. Disabled by default.
Step 8	**show vrrp** **Example:** `switch(config-if-vrrp)# show vrrp`	(Optional) Displays a summary of VRRP information.
Step 9	**copy running-config startup-config** **Example:** `switch(config-if-vrrp)# copy running-config startup-config`	(Optional) Saves this configuration change.

Configuring VRRP Interface State Tracking

Interface state tracking changes the priority of the virtual router based on the state of another interface in the switch. 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" section on page 1-9).

Note For interface state tracking to function, you must enable preemption on the interface.

Note VRRP does not support Layer 2 interface tracking.

BEFORE YOU BEGIN

Ensure that you have enabled the VRRP feature (see the "Configuring VRRP" section on page 1-7).

Ensure that you have configured an IP address on the interface (see the "Configuring IPv4 Addressing" section on page 1-8.

Ensure that you have enabled the virtual router (see the "Configuring VRRP Groups" section on page 1-8).

SUMMARY STEPS

1. **configure terminal**
2. **interface** *interface-type slot/port*
3. **no switchport**
4. **vrrp** *number*
5. **shutdown**
6. **track interface** *type number* **priority** *value*
7. **no shutdown**
8. (Optional) **show vrrp**
9. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`interface`** *`interface-type slot/port`*
	
	Example:
	`switch(config)# interface ethernet 2/1`
	`switch(config-if)#`
		Enters interface configuration mode.
		Note If this is a 10G breakout port, the *slot/port* syntax is *slot/QSFP-module/port*.

Step 3	**`no switchport`**
	
	Example:
	`switch(config-if)# no switchport`
		Configures the interface as a Layer 3 routed interface.

	Command	Purpose

Step 4	**`vrrp`** *`number`*
	`Example:`
	`switch(config-if)# vrrp 250`
	`switch(config-if-vrrp)#`
		Creates a virtual router group.

Step 5	**`shutdown`**
	`Example``:`
	`switch(config-if-vrrp)# shutdown`
	`switch(config-if-vrrp)#`
		Disables the VRRP group. Disabled by default.

Step 6	**`track interface`** *`type number`* **`priority`** *`value`*
	`Example:`
	`switch(config-if-vrrp)# track interface ethernet 2/10 priority 254`
		Enables interface priority tracking for a VRRP group. The priority range is from 1 to 254.

Step 7	**`no shutdown`**
	`Example``:`
	`switch(config-if-vrrp)# no shutdown`
	`switch(config-if-vrrp)#`
		Enables the VRRP group. Disabled by default.

Step 8	**`show vrrp`**
	`Example:`
	`switch(config-if-vrrp)# show vrrp`
		(Optional) Displays a summary of VRRP information.

Step 9	**`copy running-config startup-config`**
	`Example:`
	`switch(config-if-vrrp)# copy running-config startup-config`
		(Optional) Saves this configuration change.

Verifying the VRRP Configuration

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

Command	Purpose
show vrrp	Displays the VRRP status for all groups.
show vrrp vr *group-number*	Displays the VRRP status for a VRRP group.
show vrrp vr *number* **interface** *interface-type port* **configuration**	Displays the virtual router configuration for an interface.
show vrrp vr *number* **interface** *interface-type port* **status**	Displays the virtual router status for an interface.

Displaying VRRP Statistics

To display VRRP statistics, use the following commands:

Command	Purpose
show vrrp vr *number* **interface** *interface-type port* **statistics**	Displays the virtual router information.
show vrrp statistics	Displays the VRRP statistics.

Use the **clear vrrp vr** command to clear the IPv4 VRRP statistics for a specified interface.

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 will become the master for this group with priority 120.
  - Advertising interval is 3 seconds.
  - Preemption is enabled.
- Group 5:
  - Router B will become the master for this group with priority 200.
  - Advertising interval is 30 seconds.
  - Preemption is enabled.
- Group 100:
  - Router A will become the master for this group first because it has a higher IP address (10.1.0.2).
  - Advertising interval is the default 1 second.
  - Preemption is disabled.

Router A

```
interface ethernet 1/0
   no switchport
   ip address 10.1.0.2/16
   no shutdown
    vrrp 1
     priority 120
     authentication text cisco
     advertisement-interval 3
     address 10.1.0.10
     no shutdown
    vrrp 5
     priority 100
     advertisement-interval 30
     address 10.1.0.50
     no shutdown
    vrrp 100
     no preempt
     address 10.1.0.100
     no shutdown
```

Router B

```
interface ethernet 1/0
 no switchport
 ip address 10.2.0.1/2
 no shutdown
    vrrp 1
     priority 100
     authentication text cisco
     advertisement-interval 3
     address 10.2.0.10
     no shutdown

    vrrp 5
     priority 200
     advertisement-interval 30
     address 10.2.0.50
     no shutdown
    vrrp 100
     no preempt
     address 10.2.0.100
     no shutdown
```

Additional References

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

- Related Documents, page 1-19

Related Documents

Related Topic	Document Title
Configuring the Hot Standby Routing Protocol	Chapter 1, "Configuring HSRP"
VRRP CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Configuring Object Tracking

This chapter describes how to configure object tracking on Cisco NX-OS switches.

This chapter includes the following sections:

Information About Object Tracking

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

This section includes the following topics:

Object Tracking Overview

The object tracking feature allows you to create a tracked object that multiple clients can use to modify the client behavior when a tracked object changes. Several clients register their interest with the tracking process, track the same object, and take different actions when the object state changes.

Clients include the following features:

- Hot Standby Redundancy Protocol (HSRP)

- Virtual port channel (vPC)
- Virtual Router Redundancy Protocol (VRRP)

The object tracking monitors the status of the tracked objects and communicates any changes made to interested clients. Each tracked object is identified by a unique number that clients can use to configure the action to take when a tracked object changes state.

Cisco NX-OS tracks the following object types:

- Interface line protocol state—Tracks whether the line protocol state is up or down.
- Interface IP routing state—Tracks whether the interface has an IPv4 address and if IPv4 routing is enabled and active.
- IP route reachability—Tracks whether an Ipv4 route exists and is reachable from the local switch.

For example, you can configure HSRP to track the line protocol of the interface that connects one of the redundant routers to the rest of the network. If that link protocol goes down, you can modify the priority of the affected HSRP router.

Object Track List

An object track list allows you to track the combined states of multiple objects. Object track lists support the following capabilities:

- Boolean "and" function—Each object defined within the track list must be in an up state so that the track list object can become up.
- Boolean "or" function—At least one object defined within the track list must be in an up state so that the tracked object can become up.
- Threshold percentage—The percentage of up objects in the tracked list must be greater than the configured up threshold for the tracked list to be in the up state. If the percentage of down objects in the tracked list is above the configured track list down threshold, the tracked list is marked as down.
- Threshold weight—Assign a weight value to each object in the tracked list, and a weight threshold for the track list. If the combined weights of all up objects exceeds the track list weight up threshold, the track list is in an up state. If the combined weights of all the down objects exceeds the track list weight down threshold, the track list is in the down state.

Other entities, such as virtual Port Channels (vPCs) can use an object track list to modify the state of a vPC based on the state of the multiple peer links that create the vPC. See the *Cisco Nexus 6000 Series NX-OS Interfaces Configuration Guide, Release 6.x*, for more information on vPCs.

See the "Configuring an Object Track List with a Boolean Expression" section on page 1-6 for more information on track lists.

Virtualization Support

Object tracking supports Virtual Routing and Forwarding (VRF) instances. By default, Cisco NX-OS places you in the default VRF unless you specifically configure another VRF. By default, Cisco NX-OS tracks the route reachability state of objects in the default VRF. If you want to track objects in another VRF, you must configure the object to be a member of that VRF (see the "Configuring Object Tracking for a Nondefault VRF" section on page 1-12).

For more information, see Chapter 1, "Configuring Layer 3 Virtualization."

Licensing Requirements for Object Tracking

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Object tracking requires no license. Any feature not included in a license package is bundled with the Cisco NX-OS system images and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations

Object tracking has the following configuration guidelines and limitations:

- Supports up to 500 tracked objects.
- Supports Ethernet, subinterfaces, tunnels, port channels, loopback interfaces, and VLAN interfaces.
- Supports one tracked object per HSRP group.

Default Settings

Table 1-1 lists the default settings for object tracking parameters.

Table 1-1 Default Object Tracking Parameters

Parameters	Default
Tracked Object VRF	Member of default VRF

Configuring Object Tracking

This section includes the following topics:

- Configuring Object Tracking for an Interface, page 1-4
- Configuring Object Tracking for Route Reachability, page 1-5
- Configuring an Object Track List with a Boolean Expression, page 1-6
- Configuring an Object Track List with a Percentage Threshold, page 1-7
- Configuring an Object Track List with a Weight Threshold, page 1-8
- Configuring an Object Tracking Delay, page 1-10
- Configuring Object Tracking for a Nondefault VRF, page 1-12

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 Object Tracking for an Interface

You can configure Cisco NX-OS to track the line protocol or IPv4 routing state of an interface.

SUMMARY STEPS

1. **configure terminal**
2. **track** *object-id* **interface** *interface-type number* {**ip routing** | **line-protocol**}
3. (Optional) **show track** [*object-id*]
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**track** *object-id* **interface** *interface-type number* {**ip routing** | **line-protocol**} **Example:** switch(config)# track 1 interface ethernet 1/2 line-protocol switch(config-track#	Creates a tracked object for an interface and enters tracking configuration mode. The *object-id* range is from 1 to 500.
Step 3	**show track** [*object-id*] **Example**: switch(config-track)# show track 1	(Optional) Displays object tracking information.
Step 4	**copy running-config startup-config** **Example:** switch(config-track)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure object tracking for the line protocol state on Ethernet 1/2:

```
switch# configure terminal
switch(config)# track 1 interface ethernet 1/2 line-protocol
switch(config-track)# copy running-config startup-config
```

This example shows how to configure object tracking for the IPv4 routing state on Ethernet 1/2:

```
switch# configure terminal
switch(config)# track 2 interface ethernet 1/2 ip routing
switch(config-track)# copy running-config startup-config
```

Configuring Object Tracking for Route Reachability

You can configure Cisco NX-OS to track the existence and reachability of an IP route.

SUMMARY STEPS

1. **configure terminal**
2. **track** *object-id* **ip route** *prefix/length* **reachability**
3. (Optional) **show track** [*object-id*]
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**track** *object-id* **ip route** *prefix/length* **reachability**
	Example:
	`switch(config)# track 2 ip route 192.0.2.0/8 reachability`
	`switch(config-track)#`
		Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500. The prefix format for IP is A.B.C.D/length, where the length range is from 1 to 32.

Step 3	**show track** [*object-id*] **Example**: `switch(config-track)# show track 1`	(Optional) Displays object tracking information.
Step 4	**copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure object tracking for an IPv4 route in the default VRF.

```
switch# configure terminal
switch(config)# track 4 ip route 192.0.2.0/8 reachability
switch(config-track)# copy running-config startup-config
```

Configuring an Object Track List with a Boolean Expression

You can configure an object track list that contains multiple tracked objects. A tracked list contains one or more objects. The Boolean expression enables two types of calculation by using either "and" or "or" operators. For example, when tracking two interfaces using the "and" operator, up means that both interfaces are up, and down means that either interface is down.

SUMMARY STEPS

1. **configure terminal**
2. **track** *track-number* **list boolean** {**and** | **or**}
3. **object** *object-number* [**not**]
4. (Optional) **show track**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`track`** *`track-number`* **`list boolean`** `{`**`and`** `|` **`or`**`}`
	
	Example:
	`switch(config)# track 1 list boolean and`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a Boolean calculation. The keywords are as follows:
		• **and**—Specifies that the list is up if all objects are up, or down if one or more objects are down. For example, when tracking two interfaces, up means that both interfaces are up, and down means that either interface is down.
		• **or**—Specifies that the list is up if at least one object is up. For example, when tracking two interfaces, up means that either interface is up, and down means that both interfaces are down.
		The *track-number* range is from 1 to 500.

Step 3	**`object`** *`object-id`* `[`**`not`**`]`
	
	Example:
	`switch(config-track)# object 10`
		Adds a tracked object to the track list. The *object-id* range is from 1 to 500. The **not** keyword optionally negates the tracked object state.
		Note The example means that when object 10 is up, the tracked list detects object 10 as down.

	Command	Purpose
Step 4	**show track** **Example**: `switch(config-track)# show track`	(Optional) Displays object tracking information.
Step 5	**copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure a track list with multiple objects as a Boolean “and”:

```
switch# configure terminal
switch(config)# track 1 list boolean and
switch(config-track)# object 10
switch(config-track)# object 20 not
```

Configuring an Object Track List with a Percentage Threshold

You can configure an object track list that contains a percentage threshold. A tracked list contains one or more objects. The percentage of up objects must exceed the configured track list up percent threshold before the track list is in an up state. For example, if the tracked list has three objects, and you configure an up threshold of 60 percent, two of the objects must be in the up state (66 percent of all objects) for the track list to be in the up state.

SUMMARY STEPS

1. **configure terminal**
2. **track** *track-number* **list threshold percentage**
3. **threshold percentage up** *up-value* **down** *down-value*
4. **object** *object-number*
5. (Optional) **show track**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**track** *track-number* **list threshold percentage**
	Example:
	`switch(config)# track 1 list threshold percentage`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a configured threshold percent.
		The *track-number* range is from 1 to 500.

	Command	Purpose
Step 3	**threshold percentage up** *up-value* **down** *down-value* **Example:** `switch(config-track)# threshold percentage up 70 down 30`	Configures the threshold percent for the tracked list. The range from 0 to 100 percent.
Step 4	**object** *object-id* **Example:** `switch(config-track)# object 10`	Adds a tracked object to the track list. The *object-id* range is from 1 to 500.
Step 5	**show track** **Example**: `switch(config-track)# show track`	(Optional) Displays object tracking information.
Step 6	**copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure a track list with an up threshold of 70 percent and a down threshold of 30 percent:

```
switch# configure terminal
switch(config)# track 1 list threshold percentage
switch(config-track)# threshold percentage up 70 down 30
switch(config-track)# object 10
switch(config-track)# object 20
switch(config-track)# object 30
```

Configuring an Object Track List with a Weight Threshold

You can configure an object track list that contains a weight threshold. A tracked list contains one or more objects. The combined weight of up objects must exceed the configured track list up weight threshold before the track list is in an up state. For example, if the tracked list has three objects with the default weight of 10 each, and you configure an up threshold of 15, two of the objects must be in the up state (combined weight of 20) for the track list to be in the up state.

SUMMARY STEPS

1. **configure terminal**
2. **track** *track-number* **list threshold weight**
3. **threshold weight up** *up-value* **down** *down-value*
4. **object** *object-number* **weight** *value*
5. (Optional) **show track**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters configuration mode.
Step 2	**track** *track-number* **list threshold weight** **Example:** switch(config)# track 1 list threshold weight switch(config-track#	Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a configured threshold weight. The *track-number* range is from 1 to 500.
Step 3	**threshold weight up** *up-value* **down** *down-value* **Example:** switch(config-track)# threshold weight up 30 down 10	Configures the threshold weight for the tracked list. The range from 1 to 255.
Step 4	**object** *object-id* **weight** *value* **Example:** switch(config-track)# object 10 weight 15	Adds a tracked object to the track list. The *object-id* range is from 1 to 500. The *value* range is from 1 to 255. The default weight value is 10.
Step 5	**show track** **Example**: switch(config-track)# show track	(Optional) Displays object tracking information.
Step 6	**copy running-config startup-config** **Example:** switch(config-track)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure a track list with an up weight threshold of 30 and a down threshold of 10:

```
switch# configure terminal
switch(config)# track 1 list threshold weight
switch(config-track)# threshold weight up 30 down 10
switch(config-track)# object 10 weight 15
switch(config-track)# object 20 weight 15
switch(config-track)# object 30
```

In this example, the track list is up if object 10 and object 20 are up, and the track list goes to the down state if all three objects are down.

Configuring an Object Tracking Delay

You can configure a delay for a tracked object or an object track list that delays when the object or list triggers a stage change. The tracked object or track list starts the delay timer when a state change occurs but does not recognize a state change until the delay timer expires. At that point, Cisco NX-OS checks the object state again and records a state change only if the object or list currently has a changed state. Object tracking ignores any intermediate state changes before the delay timer expires.

For example, for an interface line-protocol tracked object that is in the up state with a 20-second donw delay, the delay timer starts when the line protocol goes down. The object is not in the down state unless the line protocol is down 20 seconds later.

You can configure independent up delay and down delay for a tracked object or track list. When you delete the delay, object tracking deletes both the up and down delay.

You can change the delay at any point. If the object or list is already counting down the delay timer from a triggered event, the new delay is computed as the following:

- If the new configuration value is less than the old configuration value, the timer starts with the new value.
- If the new configuration value is more than the old configuration value, the timer is calculated as the new configuration value minus the current timer countdown minus the old configuration value.

SUMMARY STEPS

1. **configure terminal**
2. **track** *object-id* {*parameters*}
3. **track** *track-number* **list {***parameters***}**
4. **delay** {**up** *up-time* [**down** *down-time*] | **down** *down-time* [**up** *up-time*]}
5. (Optional) **show track**
6. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**`track`** *`object-id`* `{`*`parameters`*`}`
	Example:
	`switch(config)# track 2 ip route 192.0.2.0/8 reachability`
	`switch(config-track)#`
		Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500. The prefix format for IP is A.B.C.D/length, where the length range is from 1 to 32.

Step 3	**`track`** *`track-number`* **`list`** `{`*`parameters`*`}`
	Example:
	`switch(config)# track 1 list threshold weight`
	`switch(config-track#`
		Configures a tracked list object and enters tracking configuration mode. Specifies that the state of the tracked list is based on a configured threshold weight.
		The *track-number* range is from 1 to 500.

	Command	Purpose
Step 4	**delay** {**up** *up-time* [**down** *down-time*] | **down** *down-time* [**up** *up-time*]} **Example**: switch(config-track)# delay up 20 down 30	Configures the object delay timers. The range is from 0 to 180 seconds.
Step 5	**show track** **Example**: switch(config-track)# show track 3	(Optional) Displays object tracking information.
Step 6	**copy running-config startup-config** **Example:** switch(config-track)# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to configure object tracking for a route and use delay timers:

```
switch# configure terminal
switch(config)# track 2 ip route 209.165.201.0/8 reachability
switch(config-track)# delay up 20 down 30
switch(config-track)# copy running-config startup-config
```

This example shows how to configure a track list with an up weight threshold of 30 and a down threshold of 10 with delay timers:

```
switch# configure terminal
switch(config)# track 1 list threshold weight
switch(config-track)# threshold weight up 30 down 10
switch(config-track)# object 10 weight 15
switch(config-track)# object 20 weight 15
switch(config-track)# object 30
switch(config-track)# delay up 20 down 30
```

This example shows the delay timer in the **show track** command output before and after an interface is shut down:

```
switch(config-track)# show track
Track 1
  Interface loopback1 Line Protocol
  Line Protocol is UP
  1 changes, last change 00:00:13
  Delay down 10 secs

switch(config-track)# interface loopback 1
switch(config-if)# shutdown
switch(config-if)# show track
Track 1
  Interface loopback1 Line Protocol
  Line Protocol is delayed DOWN (8 secs remaining)<------- delay timer counting down
  1 changes, last change 00:00:22
  Delay down 10 secs
```

Configuring Object Tracking for a Nondefault VRF

You can configure Cisco NX-OS to track an object in a specific VRF.

SUMMARY STEPS

1. **configure terminal**
2. **track** *object-id* **ip route** *prefix/length* **reachability**
3. **vrf member** *vrf-name*
4. (Optional) **show track** [*object-id*]
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

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

Step 2	**track** *object-id* **ip route** *prefix/length* **reachability**
	Example:
	`switch(config)# track 2 ip route 192.0.2.0/8 reachability`
	`switch(config-track)#`
		Creates a tracked object for a route and enters tracking configuration mode. The *object-id* range is from 1 to 500. The prefix format for IP is A.B.C.D/length, where the length range is from 1 to 32.

Step 3	**vrf member** *vrf-name* **Example**: `switch(config-track)# vrf member Red`	Configures the VRF to use for tracking the configured object.
Step 4	**show track** [*object-id*] **Example**: `switch(config-track)# show track 3`	(Optional) Displays object tracking information.
Step 5	**copy running-config startup-config** **Example:** `switch(config-track)# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to configure object tracking for a route and use VRF Red to look up reachability information for this object:

```
switch# configure terminal
switch(config)# track 2 ip route 209.165.201.0/8 reachability
switch(config-track)# vrf member Red
switch(config-track)# copy running-config startup-config
```

This example shows how to modify tracked object 2 to use VRF Blue instead of VRF Red to look up reachability information for this object:

```
switch# configure terminal
switch(config)# track 2
switch(config-track)# vrf member Blue
switch(config-track)# copy running-config startup-config
```

Verifying the Object Tracking Configuration

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

Command	Purpose
show track [*object-id*] [**brief**]	Displays the object tracking information for one or more objects.
show track [*object-id*] **interface** [**brief**]	Displays the interface-based object tracking information.
show track [*object-id*] **ip route** [**brief**]	Displays the IPv4 route-based object tracking information.

Configuration Examples for Object Tracking

This example shows how to configure object tracking for route reachability and use VRF Red to look up reachability information for this route:

```
switch# configure terminal
switch(config)# track 2 ip route 209.165.201.0/8 reachability
switch(config-track)# vrf member Red
switch(config-track)# copy running-config startup-config
```

Related Topics

See the following topics for information related to object tracking:

- Chapter 1, “Configuring Layer 3 Virtualization”
- Chapter 1, “Configuring HSRP”

Additional References

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

- Related Documents, page 1-14
- Standards, page 1-14

Related Documents

Related Topic	Document Title
Object Tracking CLI commands	*Cisco Nexus 5000 Series Command Reference, Cisco NX-OS Releases 4.x, 5.x*

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

APPENDIX A

IETF RFCs supported by Cisco NX-OS Unicast Features, Release 6.x

This appendix lists the IETF RFCs supported in Cisco NX-OS Release 6.x.

BGP RFCs

RFCs	Title
RFC 1997	*BGP Communities Attribute*
RFC 2385	*Protection of BGP Sessions via the TCP MD5 Signature Option*
RFC 2439	*BGP Route Flap Damping*
RFC 2519	*A Framework for Inter-Domain Route Aggregation*
RFC 2858	*Multiprotocol Extensions for BGP-4*
RFC 3065	*Autonomous System Confederations for BGP*
RFC 3392	*Capabilities Advertisement with BGP-4*
RFC 4271	*A Border Gateway Protocol 4 (BGP-4)*
RFC 4273	*Definitions of Managed Objects for BGP-4*
RFC 4456	*BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)*
RFC 4486	*Subcodes for BGP Cease Notification Message*
RFC 4893	*BGP Support for Four-octet AS Number Space*
RFC 5004	*Avoid BGP Best Path Transitions from One External to Another*
draft-ietf-idr-bgp4-mib-15.txt	*BGP4-MIB*

First-Hop Redundancy Protocols RFCs

RFCs	Title
RFC 2281	*Hot Standby Redundancy Protocol*
RFC 3768	*Virtual Router Redundancy Protocol*

IP Services RFCs

RFCs	Title
RFC 786	*UDP*
RFC 791	*IP*
RFC 792	*ICMP*
RFC 793	*TCP*
RFC 826	*ARP*
RFC 1027	*Proxy ARP*
RFC 1591	*DNS Client*
RFC 1812	*IPv4 routers*

IPv6 RFCs

RFCs	Title
RFC 1981	*Path MTU Discovery for IP version 6*
RFC 2373	IP Version 6 Addressing Architecture
RFC 2374	*An Aggregatable Global Unicast Address Format*
RFC 2460	*Internet Protocol, Version 6 (IPv6) Specification*
RFC 2461	*Neighbor Discovery for IP Version 6 (IPv6)*
RFC 2462	*IPv6 Stateless Address Autoconfiguration*
RFC 2463	*Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*
RFC 2464	*Transmission of IPv6 Packets over Ethernet Networks*
RFC 3152	*Delegation of IP6.ARPA*
RFC 3162	*RADIUS and IPv6*
RFC 3513	*Internet Protocol Version 6 (IPv6) Addressing Architecture*
RFC 3596	*DNS Extensions to Support IP version 6*
RFC 4193	*Unique Local IPv6 Unicast Addresses*
RFC 5095	*Deprecation of Type 0 of Routing Headers in IPv6*

OSPF RFCs

RFCs	Title
RFC 2328	*OSPF Version 2*
RFC 3101	*The OSPF Not-So-Stubby Area (NSSA) Option*

RFCs	Title
RFC 2370	*The OSPF Opaque LSA Option*
RFC 3137	*OSPF Stub Router Advertisement*

RIP RFCs

RFCs	Title
RFC 2453	*RIP Version 2*
RFC 2082	*RIP-2 MD5 Authentication*

GLOSSARY

A

ABR See area border router.

address family A specific type of network addressing supported by a routing protocol. Examples include IPv4 unicast and IPv4 multicast.

adjacency Two OSPF routers that have compatible configurations and have synchronized their link-state databases.

administrative distance A rating of the trustworthiness of a routing information source. In general, the higher the value, the lower the trust rating.

area A logical division of routers and links within an OSPF domain that creates separate subdomains. LSA flooding is contained within an area.

area border router A router that connects one OSPF area to another OSPF area.

ARP Address Resolution Protocol. ARP discovers the MAC address for a known IPv4 address.

AS See autonomous system.

ASBR See autonomous system border router.

attributes Properties of a route that are sent in BGP UPDATE messages. These attributes include the path to the advertised destination as well as configurable options that modify the best path selection process.

autonomous system A network controlled by a single technical administration entity.

autonomous system border router A router that connect a an OSPF autonomous system to an external autonomous system.

AVF Active virtual forwarder. A gateway within a GLBP group elected to forward traffic for a specified virtual MAC address.

AVG Active virtual gateway. One virtual gateway within a GLBP group is elected as the active virtual gateway and is responsible for the operation of the protocol.

B

backup designated router See BDR.

bandwidth The available traffic capacity of a link.

BDR Backup designated router. An elected router in a multi-access OSPF network that acts as the backup if the designated router fails. All neighbors form adjacencies with the backup designated router (BDR) as well as the designated router.

BGP Border Gateway Protocol. BGP is an interdomain or exterior gateway protocol.

BGP peer A remote BGP speaker that is an established neighbor of the local BGP speaker.

BGP speaker BGP-enabled router.

C

communication cost Measure of the operating cost to route over a link.

converged The point at which all routers in a network have identical routing information.

convergence See converged.

D

dead interval The time within which an OSPF router must receive a Hello packet from an OSPF neighbor. The dead interval is usually a multiple of the hello interval. If no Hello packet is received, the neighbor adjacency is removed.

default gateway A router to which all unroutable packets are sent. Also called the router of last resort.

delay The length of time required to move a packet from the source to the destination through the internetwork.

designated router See DR.

DHCP Dynamic Host Control Protocol.

Diffusing Update Algorithm See DUAL.

distance vector Defines routes by distance (for example, the number of hops to the destination) and direction (for example, the next-hop router) and then broadcasts to the directly connected neighbor routers.

DNS client Domain Name System client. Communicates with DNS server to translate a host name to an IP address.

DR Designated router. An elected router in a multi-access OSPF network that sends LSAs on behalf of all its adjacent neighbors. All neighbors establish adjacency with only the designated router and the backup designated router.

DUAL Diffusing Update Algorithm. EIGRP algorithm used to select optimal routes to a destination.

E

eBGP External Border Gateway Protocol (BGP). Operates between external systems.

EIGRP Enhanced Interior Gateway Protocol. A Cisco routing protocol that uses the Diffusing Update Algorithm to provide fast convergence and minimized bandwidth utilization.

F

feasible distance The lowest calculated distance to a network destination in EIGRP. The feasibility distance is the sum of the advertised distance from a neighbor plus the cost of the link to that neighbor.

feasible successor Neighbors in EIGRP that advertise a shorter distance to the destination than the current feasibility distance.

FIB Fowarding Information Base. The forwarding table on each module that is used to make the Layer 3 forwarding decisions per packet.

G

gateway A switch or router that forwards Layer 3 traffic from a LAN to the rest of the network.

H

hello interval The configurable time between each hello packet sent by an OSPF or EIGRP router.

hello packet A special message used by OSPF or IS-IS to discover neighbors. Also acts as a keepalive messages between established neighbors.

hold time In BGP, the maximum time limit allowed in BGP between UPDATE or KEEPALIVE messages. If this time is exceeded, the TCP connection between the BGP peers is closed.

In EIGRP, the maximum time allowed between EIGRP hello messages. If this time is exceeded, the neighbor is declared unreachable.

hop count The number of routers that can be traversed in a route. Used by RIP.

I

iBGP Internal Border Gateway Protocol (BGP). Operates within an autonomous system.

ICMP

IETF RFCs Internet Engineering Task Force Request for Comments.

IGP Interior Gateway Protocol. Used between routers within the same autonomous system.

instance An independent, configurable entity, typically a protocol.

IP tunnels

IPv4 Internet Protocol version 4.

K

keepalive A special message sent between routing peers to verify and maintain communications between the pair.

L

link cost An arbitrary number configured on an OSPF interface which is in shortest path first calculations.

link-state Shares information about a link, link cost to neighboring routers.

link-state advertisement See LSA.

LSA Link-state advertisement. An OSPF message to share information on the operational state of a link, link cost, and other OSPF neighbor information.

link-state database OSPF database of all LSAs received. OSPF uses this database to calculate the best path to each destination in the network.

link-state refresh The time that OSPF floods the network with LSAs to ensure all OSPF routers have the same information.

load The degree to which a network resource, such as a router, is busy.

load balancing The distribution of network traffic across multiple paths to a given destination.

M

message digest A one-way hash applied to a message using a shared password and appended to the message to authenticate the message and ensure the message has not been altered in transit.

metric A standard of measurement, such as the path bandwidth, that is used by routing algorithms to determine the optimal path to a destination.

MD5 authentication digest A cryptographic construction that is calculated based on an authentication key and the original message and sent along with the message to the destination. Allows the destination to determine the authenticity of the sender and guarantees that the message has not been tampered with during transmission.

MTU Maximum transmission unit. The largest packet size that a network link will transmit without fragmentation.

N

network layer reachability information BGP network layer reachability information (NRLI). Contains the a list of network IP addresses and network masks for networks that are reachable from the advertising BGP peer.

next hop The next router that a packet is sent to on its way to the destination address.

NSSA Not-So-Stubby-Area. Limits AS external LSAs in an OSPF area.

O

OSPF Open Shortest Path First. An IETF link-state protocol. OSPFv2 supports IPv4.

P

path length Sum of all link costs or the hop count that a packet experiences when routed from the source to the destination.

R

redistribution One routing protocol accepts route information from another routing protocol and advertises it in the local autonomous system.

Reliable Transport Protocol Responsible for guaranteed, ordered delivery of EIGRP packets to all neighbors.

reliability The dependability (usually described in terms of the bit-error rate) of each network link.

RIB Routing Information Base. Maintains the routing table with directly connected routes, static routes, and routes learned from dynamic unicast routing protocols.

routing information base See RIB.

route map A construct used to map a route or packet based on match criteria and optionally alter the route or packet based on set criteria. Used in route redistribution.

route summarization A process that replaces a series of related, specific routes in a route table with a more generic route.

router ID A unique identifier used by routing protocols. If not manually configured, the routing protocol selects the highest IP address configured on the system.

S

SPF algorithm Shortest Path First algorithm. Dijkstra’s algorithm used by OSPF to determine the shortest route through a network to a particular destination.

split horizon Routes learned from an interface are not advertised back along the interface they were learned on, preventing the router from seeing its own route updates.

split horizon with poison reverse Routes learned from an interface are set as unreachable and advertised back along the interface they were learned on, preventing the router from seeing its own route updates.

static route A manually configured route.

stub area An OSPF area that does not allow AS External (type 5) LSAs.

stub router A router that has no direct connection to the main network and which routes to that network using a known remote router.

SVI Switched Virtual Interface.

U

UFIB Unicast IPv4 forwarding information base.

URIB Unicast IPv4 routing information base. The unicast routing table that gathers information from all routing protocols and updates the forwarding information base for each module.

V

virtualization A method of making a physical entity act as multiple, independent logical entities.

VRF Virtual Routing and Forwarding. A method used to create separate, independent Layer 3 entities within a system.

VRRP Virtual Router Redundancy Protocol.

INDEX