Configuring IP Routing Protocol-Independent Features

Last Updated: April 30, 2012

The Configuring IP Routing Protocol-Independent Features module describes how to configure IP routing protocol-independent features. For a complete description of the IP routing protocol-independent commands in this chapter, see the *Cisco IOS IP Routing: Protocol-Independent Command Reference*. To locate documentation of other commands in this chapter, use the command reference master index or search online.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About Configuring IP Routing Protocol-Independent Features

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Variable-Length Subnet Masks

Enhanced Interior Gateway Routing Protocol (EIGRP), Intermediate System-to-Intermediate System (IS-IS), Open Shortest Path First (OSPF), Routing Information Protocol (RIP) Version 2, and static routes support variable-length subnet masks (VLSMs). With VLSMs, you can use different masks for the same network number on different interfaces, which allows you to conserve IP addresses and more efficiently use available address space. However, using VLSMs also presents address assignment challenges for the network administrator and ongoing administrative challenges.

Refer to RFC 1219 for detailed information about VLSMs and how to correctly assign addresses.

Note Consider your decision to use VLSMs carefully. You can easily make mistakes in address assignments and you will generally find it more difficult to monitor your network using VLSMs.

Note The best way to implement VLSMs is to keep your existing addressing plan in place and gradually migrate some networks to VLSMs to recover address space. See the GUID-D922BE86-09DF-47A2-A0B5-31507877CEF0 for an example of using VLSMs.

Static Routes

Static routes are user-defined routes that cause packets moving between a source and a destination to take a specified path. Static routes can be important if the router cannot build a route to a particular destination. They are also useful for specifying a gateway of last resort to which all unroutable packets will be sent.

To configure a static route, use the **ip route** *prefix mask*{*ip-address*| *interface-type interface-number*[*ip-address*]}[*distance*] [*name*] [**permanent** | **track** *number*] [**tag** *tag*] command in global configuration mode.

See the Controlling the Advertising of Routes in Routing Updates, page 19 for an example of configuring static routes.

Static routes remains in the router configuration until you remove them (using the **no** form of the **ip route** global configuration command). However, you can override static routes with dynamic routing information through prudent assignment of administrative distance values. Each dynamic routing protocol has a default administrative distance, as listed in the table below. If you would like a static route to be overridden by information from a dynamic routing protocol, simply ensure that the administrative distance of the static route is higher than that of the dynamic protocol.

Table 1 Dynamic Routing Protocol Default Administrative Distances

Route Source	Default Distance
Connected interface	0
Static route	1
EIGRP summary route	5
External BGP	20
Internal EIGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
EGP	140
ODR	160
External EIGRP	170
Internal BGP	200
Unknown	255

Static routes that point to an interface will be advertised via RIP, EIGRP, and other dynamic routing protocols, regardless of whether **redistribute static** router configuration commands were specified for those routing protocols. These static routes are advertised because static routes that point to an interface are considered in the routing table to be connected and hence lose their static nature. However, if you define a static route to an interface that is not one of the networks defined in a **network** command, no dynamic routing protocols will advertise the route unless a **redistribute static** command is specified for these protocols.

When an interface goes down, all static routes through that interface are removed from the IP routing table. Also, when the software can no longer find a valid next hop for the address specified as the address of the forwarding router in a static route, the static route is removed from the IP routing table.

Default Routes

A router might not be able to determine the routes to all other networks. To provide complete routing capability, the common practice is to use some routers as smart routers and give the remaining routers default routes to the smart router. (Smart routers have routing table information for the entire internetwork.) These default routes can be passed along dynamically, or can be configured into the individual routers.

Most dynamic interior routing protocols include a mechanism for causing a smart router to generate dynamic default information that is then passed along to other routers.

Subnet Zero and All-Ones Subnet IP Addressing

When a network address is subnetted, the first subnet obtained is called subnet zero.

Consider a Class B address 172.16.0.0. By default, a Class B address has 16 bits reserved for representing the host portion, thus allowing 65534 (2^{16}–2) valid host addresses. If network 172.16.0.0/16 is subnetted by borrowing three bits from the host portion, eight (2^3) subnets are obtained. The table below shows the subnets obtained by subnetting the address 172.16.0.0, the resulting subnet mask, the corresponding broadcast address, and the range of valid host addresses.

Table 2 Subnets of 172.16.0.0/16

Subnet Address	Subnet Mask	Broadcast Address	Valid Host Addresses Range
172.16.0.0	255.255.224.0	172.16.31.255	172.16.0.1 to 172.16.31.254
172.16.32.0	255.255.224.0	172.16.63.255	172.16.32.1 to 172.16.63.254
172.16.64.0	255.255.224.0	172.16.95.255	172.16.64.1 to 172.16.95.254
172.16.96.0	255.255.224.0	172.16.127.255	172.16.96.1 to 172.16.127.254
172.16.128.0	255.255.224.0	172.16.159.255	172.16.128.1 to 172.16.159.254
172.16.160.0	255.255.224.0	172.16.191.255	172.16.160.1 to 172.16.191.254
172.16.192.0	255.255.224.0	172.16.223.255	172.16.192.1 to 172.16.223.254
172.16.224.0	255.255.224.0	172.16.255.255	172.16.224.1 to 172.16.255.254

In the above table, the first subnet (subnet 172.16.0.0) is called subnet zero.

The class of the network subnetted and the number of subnets obtained after subnetting have no role in determining subnet zero. It is the first subnet obtained when subnetting the network address. Also, when you write the binary equivalent of the subnet zero address, all the subnet bits (bits 17, 18, and 19 in this case) are zeros. Subnet zero is also known as the all-zeros subnet.

When a network address is subnetted, the last subnet obtained is called the all-ones subnet.

With reference to the above table, the last subnet obtained when you subnet network 172.16.0.0 (subnet 172.16.224.0/19) is called the all-ones subnet.

The class of the network subnetted and the number of subnets obtained after subnetting have no role in determining the all-ones subnet. Also, when you write the binary equivalent of the all-ones subnet, all the subnet bits (bits 17, 18, and 19 in this case) are ones, hence the name.

- Problems with Subnet Zero and the All-Ones Subnet, page 4

Problems with Subnet Zero and the All-Ones Subnet

According to RFC 950, “It is useful to preserve and extend the interpretation of these special (network and broadcast) addresses in subnetted networks. This means the values of all zeros and all ones in the subnet field should not be assigned to actual (physical) subnets.” Therefore, network engineers who had to calculate the number of subnets obtained by borrowing three bits would calculate 2^3–2 (6) and not 2^3 (8). The –2 ensures that subnet zero and the all-ones subnet are not used.

- Subnet Zero, page 5

- All-Ones, page 5

Subnet Zero

Using subnet zero may lead to the creation of a network and a subnet with indistinguishable addresses.

With reference to the above table, if you calculate the subnet address for 172.16.1.10, the answer you arrive at is subnet 172.16.0.0 (subnet zero). Note that this subnet address is identical to the network address 172.16.0.0, which was subnetted in the first place. So whenever you perform subnetting, you get a network and a subnet (subnet zero) with indistinguishable addresses.

Depending on the release, your Cisco IOS software, by default, may not allow an IP address belonging to subnet zero to be configured on an interface. However, some releases allow the use of the **ip subnet-zero** command in global configuration mode to overcome this restriction. Some releases have the **ip subnet-zero** command enabled by default, and these releases allow you to configure the **no ip subnet-zero** command to restrict the use of subnet zero addresses.

All-Ones

The use of the all-ones subnet for addressing may lead to the creation of a network and a subnet with identical broadcast addresses.

With reference to above table, the broadcast address for the last subnet (subnet 172.16.224.0) is 172.16.255.255. This address is identical to the broadcast address of the network 172.16.0.0, which was subnetted in the first place. So whenever you perform subnetting, you get a network and a subnet (all-ones subnet) with identical broadcast addresses. You can configure the address 172.16.230.1/19 on a router, but if that is done, you can no longer differentiate between a local subnet broadcast (172.16.255.255 [/19]) and the Class B broadcast (172.16.255.255[/16]).

Despite the inherent confusion created by the use of subnet zero and all-ones subnet, the entire address space including subnet zero and the all-ones subnet have always been usable. The use of subnet zero is allowed in some Cisco IOS software releases. You can use subnet zero in these releases by entering the **ip subnet-zero** global configuration command. Today, the use of subnet zero and the all-ones subnet is generally accepted and most vendors support their use. However, on certain networks, particularly the ones using legacy software, the use of subnet zero and the all-ones subnet can lead to problems.

Maximum Number of Paths

By default, most IP routing protocols install a maximum of four parallel routes in a routing table. Static routes always install six routes. The exception is BGP, which by default allows only one path (the best path) to a destination. However, BGP can be configured to use equal and unequal cost multipath load sharing. See the "BGP Multipath Load Sharing for Both eBGP and iBGP in an MPLS-VPN" feature of the *Cisco IOS IP Routing: BGP Configuration Guide* for more information.

The number of parallel routes that you can configure to be installed in the routing table is dependent on the installed version of Cisco IOS software. To change the maximum number of parallel paths allowed, use the **maximum-paths** *number-paths* command in router configuration mode.

Multi-Interface Load Splitting

Multi-interface load splitting allows you to efficiently control traffic that travels across multiple interfaces to the same destination. The **traffic-share min** router configuration command specifies that if multiple paths are available to the same destination, only paths with the minimum metric will be installed in the routing table. The number of paths allowed is never more than six. For dynamic routing protocols, the

number of paths is controlled by the **maximum-paths** router configuration command. The static route source can always install six paths. If more paths are available, the extra paths are discarded. If some installed paths are removed from the routing table, pending routes are added automatically.

When the **traffic-share min** command is used with the **across-interfaces** keyword, an attempt is made to use as many different interfaces as possible to forward traffic to the same destination. When the maximum path limit has been reached and a new path is installed, the router compares the installed paths. For example, if path X references the same interface as path Y and the new path uses a different interface, path X is removed and the new path is installed.

To configure traffic that is distributed among multiple routes of unequal cost for equal cost paths across multiple interfaces, use the **traffic-share min across-interfaces** command in router configuration mode.

Routing Information Redistribution

In addition to running multiple routing protocols simultaneously, Cisco IOS software can be configured to redistribute information from one routing protocol to another. For example, you can configure a router to readvertise EIGRP-derived routes using RIP, or to readvertise static routes using the EIGRP protocol. Redistribution from one routing protocol to another can be configured in all of the IP-based routing protocols.

You also can conditionally control the redistribution of routes between routing domains by configuring route maps between the two domains. A route map is a route/ packet filter that is configured with permit and deny statements, match and set clauses, and sequence numbers.

The following four tables list tasks associated with route redistribution. Although redistribution is a protocol-independent feature, some of the **match** and **set** commands are specific to a particular protocol.

To define a route map for redistribution, use the **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] command in global configuration mode.

One or more **match** commands and one or more **set** commands are configured in route map configuration mode. If there are no **match** commands, then everything matches. If there are no **set** commands, then no set action is performed.

To define conditions for redistributing routes from one routing protocol into another, see the Redistributing Routing Information, page 13

See the "Connecting to a Service Provider Using External BGP" module of the *Cisco IOS IP Routing: BGP Configuration Guide* for examples of BGP route map configuration tasks and configuration examples. See the "Configuring BGP Cost Community" feature of the *Cisco IOS IP Routing: BGP Configuration Guide* for examples of BGP communities and route maps.

The metrics of one routing protocol do not necessarily translate into the metrics of another. For example, the RIP metric is a hop count and the EIGRP metric is a combination of five metric values. In such situations, a dynamic metric is assigned to the redistributed route. Redistribution in these cases should be applied consistently and carefully in conjunction with inbound filtering to avoid routing loops.

Removing options that you have configured for the **redistribute** command requires careful use of the **no** form of the **redistribute** command to ensure that you obtain the result that you are expecting.

Supported Metric Translations

This section describes supported automatic metric translations between the routing protocols. The following descriptions assume that you have not defined a default redistribution metric that replaces metric conversions:

- RIP can automatically redistribute static routes. It assigns static routes a metric of 1 (directly connected).
- BGP does not normally send metrics in its routing updates.
- EIGRP can automatically redistribute static routes from other EIGRP-routed autonomous systems as long as the static route and any associated interfaces are covered by an EIGRP network statement. EIGRP assigns static routes a metric that identifies them as directly connected. EIGRP does not change the metrics of routes derived from EIGRP updates from other autonomous systems.

Note Any protocol can redistribute routes from other routing protocols as long as a default metric is configured.

Protocol Differences in Implementing the no redistribute Command

Caution Removing options that you have configured for the **redistribute** command requires careful use of the **no** form of the **redistribute** command to ensure that you obtain the result that you are expecting. In most cases, changing or disabling any keyword will not affect the state of other keywords.

It is important to understand that different protocols implement the **no** version of the **redistribute** command differently:

- In BGP, OSPF, and RIP configurations, the **no redistribute** command removes only the specified keywords from the **redistribute** commands in the running configuration. They use the *subtractive keyword* method when redistributing from other protocols. For example, in the case of BGP, if you configure **no redistribute static route-map interior**, only the route map is removed from the redistribution, leaving **redistribute static** in place with no filter.
- The **no redistribute isis** command removes the IS-IS redistribution from the running configuration. IS-IS removes the entire command, regardless of whether IS-IS is the redistributed or redistributing protocol.
- EIGRP used the subtractive keyword method prior to EIGRP component version rel5. Starting with EIGRP component version rel5, the **no redistribute** command removes the entire **redistribute** command when redistributing from any other protocol.

Default Passive Interfaces

In Internet service provider (ISP) and large enterprise networks, many of the distribution routers have more than 200 interfaces. Before the introduction of the Default Passive Interface feature, there were two possibilities for obtaining routing information from these interfaces:

- Configure a routing protocol such as OSPF on the backbone interfaces and redistribute connected interfaces.
- Configure the routing protocol on all interfaces and manually set most of them as passive.

Network operators may not always be able to summarize type 5 link-state advertisements (LSAs) at the router level where redistribution occurs, as in the first possibility. Thus, a large number of type 5 LSAs can be flooded over the domain.

In the second possibility, large type 1 LSAs might be flooded into the area. The Area Border Router (ABR) creates type 3 LSAs, one for each type 1 LSA, and floods them to the backbone. It is possible, however, to have unique summarization at the ABR level, which will inject only one summary route into the backbone, thereby reducing processing overhead.

The prior solution to this problem was to configure the routing protocol on all interfaces and manually set the **passive-interface** router configuration command on the interfaces where adjacency was not desired. But in some networks, this solution meant coding 200 or more passive interface statements. With the Default Passive Interface feature, this problem is solved by allowing all interfaces to be set as passive by default using a single **passive-interface default** command, then configuring individual interfaces where adjacencies are desired using the **no passive-interface** command.

Thus, the Default Passive Interface feature simplifies the configuration of distribution routers and allows the network manager to obtain routing information from the interfaces in large ISP and enterprise networks.

To set all interfaces as passive by default and then activate only those interfaces that must have adjacencies set, see the Configuring Default Passive Interfaces, page 18.

Sources of Routing Information Filtering

Filtering sources of routing information prioritizes routing information from different sources because some pieces of routing information may be more accurate than others. An administrative distance is a rating of the trustworthiness of a routing information source, such as an individual router or a group of routers. In a large network, some routing protocols and some routers can be more reliable than others as sources of routing information. Also, when multiple routing processes are running in the same router for IP, it is possible for the same route to be advertised by more than one routing process. By specifying administrative distance values, you enable the router to intelligently discriminate between sources of routing information. The router will always pick the route whose routing protocol has the lowest administrative distance.

To filter sources of routing information, see the Filtering Sources of Routing Information, page 20.

There are no general guidelines for assigning administrative distances because each network has its own requirements. You must determine a reasonable matrix of administrative distances for the network as a whole. Sources of Routing Information Filtering, page 8 shows the default administrative distance for various routing information sources.

For example, consider a router using EIGRP and RIP. Suppose you trust the EIGRP-derived routing information more than the RIP-derived routing information. In this example, because the default EIGRP administrative distance is lower than the default RIP administrative distance, the router uses the EIGRP-derived information and ignores the RIP-derived information. However, if you lose the source of the EIGRP-derived information (because of a power shutdown at the source network, for example), the router uses the RIP-derived information until the EIGRP-derived information reappears.

For an example of filtering on sources of routing information, see the section GUID-4B8856D5-89A7-4E16-86BB-8E028E2F8090.

Note You can also use administrative distance to rate the routing information from routers that are running the same routing protocol. This application is generally discouraged if you are unfamiliar with this particular use of administrative distance, because it can result in inconsistent routing information, including forwarding loops.

Note The weight of a route can no longer be set with the **distance** command. To set the weight for a route, use a route map.

Policy-Based Routing

Policy-based routing is a more flexible mechanism for routing packets than destination routing. It is a process whereby the router puts packets through a route map before routing them. The route map determines which packets are routed to which router next. You might enable policy-based routing if you want certain packets to be routed some way other than the obvious shortest path. Possible applications for policy-based routing are to provide equal access, protocol-sensitive routing, source-sensitive routing, routing based on interactive versus batch traffic, and routing based on dedicated links.

To enable policy-based routing, you must identify which route map to use for policy-based routing and create the route map. The route map itself specifies the match criteria and the resulting action if all of the match clauses are met. These steps are described in the following task tables.

To enable policy-based routing on an interface, indicate which route map the router should use by using the following command in interface configuration mode. A packet arriving on the specified interface will be subject to policy-based routing except when its destination IP address is the same as the IP address of the router's interface. To disable fast switching of all packets arriving on this interface use the **ip policy route-map** *map-tag*command in interface configuration mode.

To define the route map to be used for policy-based routing, use the **route-map** *map-tag* [**permit** | **deny**] [*sequence-number*] command in global configuration mode.

To define the criteria by which packets are examined to learn if they will be policy-based routed, use either **match length** *minimum-length maximum-length*commandor **match ip address**{*access-list-number* | *access-list-name*} [*access-list-number* | *access-list-name*] command or both in route map configuration mode. No match clause in the route map indicates all packets.

To set the precedence and specify where the packets that pass the match criteria are output, see Configuring precedence for policy-based routed packets, page 20.

The precedence setting in the IP header determines whether, during times of high traffic, the packets will be treated with more or less precedence than other packets. By default, the Cisco IOS software leaves this value untouched; the header remains with the precedence value that it had.

The precedence bits in the IP header can be set in the router when policy-based routing is enabled. When the packets containing those headers arrive at another router, the packets are ordered for transmission according to the precedence set, if the queueing feature is enabled. The router does not honor the precedence bits if queueing is not enabled; the packets are sent in FIFO order.

You can change the precedence setting, using either a number or name. The names came from RFC 791, but are evolving. You can enable other features that use the values in the **set ip precedence** route map configuration command to determine precedence. The table below lists the possible numbers and their corresponding name, from least important to most important.

Table 3 IP Precedence Values

Number	Name
0	routine

Number	Name
1	priority
2	immediate
3	flash
4	flash-override
5	critical
6	internet
7	network

The **set** commands can be used with each other. They are evaluated in the order shown in the previous task table. A usable next hop implies an interface. Once the local router finds a next hop and a usable interface, it routes the packet.

To display the cache entries in the policy route cache, use the **show ip cache policy** command.

For information about setting the precedence and specifying where the packets that pass the match the criteria for policy-based routing are output, see the Configuring precedence for policy-based routed packets, page 20.

For information about configure QoS Policy Propagation via BGP, see the Configuring QoS Policy Propagation via BGP, page 21.

See the GUID-95C82939-D3D7-4EA6-8484-7B8906E37D39 for an example of policy routing.

- Fast-Switched Policy Routing, page 10
- Local Policy Routing, page 11
- NetFlow Policy Routing, page 11

Fast-Switched Policy Routing

IP policy routing can also be fast switched. Prior to fast-switched policy routing, policy routing could only be process switched, which meant that on most platforms, the switching rate was approximately 1000 to 10,000 packets per second. Such rates were not fast enough for many applications. Users that need policy routing to occur at faster speeds can now implement policy routing without slowing down the router.

Fast-switched policy routing supports all of the **match** commands and most of the **set** commands, except for the following restrictions:

- The **set ip default** command is not supported.
- The **set interface** command is supported only over point-to-point links, unless a route cache entry exists using the same interface specified in the **set interface** command in the route map. Also, at the process level, the routing table is consulted to determine if the interface is on a reasonable path to the destination. During fast switching, the software does not make this check. Instead, if the packet matches, the software blindly forwards the packet to the specified interface.

Policy routing must be configured before you configure fast-switched policy routing. Fast switching of policy routing is disabled by default. To have policy routing be fast switched, use the **ip route-cache policy**command in interface configuration mode.

Local Policy Routing

Packets that are generated by the router are not normally policy-routed. To enable local policy routing for such packets, indicate which route map the router should use by using the following command in global configuration mode. All packets originating on the router will then be subject to local policy routing. To identify the route map to use for local policy routing, use the **ip local policy route-map** *map-tag*command in global configuration mode.

Use the **show ip local policy** command to display the route map used for local policy routing, if one exists.

NetFlow Policy Routing

NetFlow policy routing (NPR) integrates policy routing, which enables traffic engineering and traffic classification, with NetFlow services, which provide billing, capacity planning, and monitoring information on real-time traffic flows. IP policy routing now works with Cisco Express Forwarding (CEF), distributed CEF (dCEF), and NetFlow.

As quality of service (QoS) and traffic engineering become more popular, so does interest in the ability of policy routing to selectively set IP Precedence and type of service (ToS) bits (based on access lists and packet size), thereby routing packets based on predefined policy. It is important that policy routing work well in large, dynamic routing environments. Hence, distributed support allows customers to leverage their investment in distributed architecture.

NetFlow policy routing leverages the following technologies:

- CEF, which looks at a Forwarding Information Base (FIB) instead of a routing table when switching packets, to address maintenance problems of a demand caching scheme.
- dCEF, which addresses the scalability and maintenance problems of a demand caching scheme.
- NetFlow, which provides accounting, capacity planning, and traffic monitoring capabilities.

Following are NPR benefits:

- NPR takes advantage of the new switching services. CEF, dCEF, and NetFlow can now use policy routing.
- Now that policy routing is integrated into CEF, policy routing can be deployed on a wide scale and on high-speed interfaces.

Following are NPR restrictions:

- NPR is only available on Cisco IOS platforms that support CEF.
- Distributed FIB-based policy routing is only available on platforms that support dCEF.
- The **set ip next-hop verify-availability** command is not supported in dCEF because dCEF does not support the Cisco Discovery Protocol (CDP) database.

In order for NetFlow policy routing to work, the following features must already be configured:

- CEF, dCEF, or NetFlow
- Policy routing

To configure CEF, or dCEF, refer to the "Cisco Express Forwarding Overview" chapter of the *Cisco IOS IP Switching Configuration Guide*. To configure NetFlow, refer to the "Cisco IOS NetFlow Overview" chapter of the *Cisco IOS NetFlow Configuration Guide*.

NPR is the default policy routing mode. No additional configuration tasks are required to enable policy routing in conjunction with CEF, dCEF, or NetFlow. As soon as one of these features is turned on, packets are automatically subject to policy routing in the appropriate switching path.

There is one new, optional configuration command (**set ip next-hop verify-availability**). This command has the following restrictions:

- It can cause some performance degradation due to CDP database lookup overhead per packet.
- CDP must be enabled on the interface.
- The directly connected next hop must be a Cisco device with CDP enabled.
- The command will not work with dCEF configurations, due to the dependency of the CDP neighbor database.

It is assumed that policy routing itself is already configured.

If the router is policy routing packets to the next hop and the next hop happens to be down, the router will try unsuccessfully to use Address Resolution Protocol (ARP) for the next hop (which is down). This behavior can continue indefinitely.

To prevent this situation from occurring, you can configure the router to first verify that the next hop, using a route map, are CDP neighbors of the router before routing to that next hop.

This task is optional because some media or encapsulations do not support CDP, or it may not be a Cisco device that is sending the router traffic.

To configure the router to verify that the next hop is a CDP neighbor before the router tries to policy-route to it, use the **set ip next-hop verify-availability** command in route map configuration mode.

If the command shown is set and the next hop is not a CDP neighbor, the router looks to the subsequent next hop, if there is one. If there is none, the packets are simply not policy-routed.

If the command shown is not set, the packets are either policy-routed or remain forever unrouted.

If you want to selectively verify availability of only some next hops, you can configure different route-map entries (under the same route-map name) with different criteria (using access list matching or packet size matching), and use the **set ip next-hop verify-availability** configuration command selectively.

Typically, you would use existing policy routing and NetFlow **show** commands to monitor these features. For more information on these **show** commands, refer to the *Cisco IOS IP Routing: Protocol Independent Command Reference* for policy routing commands and the appropriate chapter of the *Cisco IOS IP NetFlow Command Reference* for NetFlow commands.

To display the route-map Inter Processor Communication (IPC) message statistics in the Route Processor (RP) or Versatile Interface Processor (VIP), use the **show route-map ipc** command in EXEC mode.

QoS Policy Propogation via BGP

The QoS Policy Propagation via BGP feature allows you to classify packets by IP precedence based on BGP community lists, BGP autonomous system paths, and access lists. After a packet has been classified, you can use other Quality of Service features such as committed access rate (CAR) and Weighted Random Early Detection (WRED) to specify and enforce policies to fit your business model.

Before you configure policy propagation via BGP, perform the following basic tasks:

- Configure BGP and Cisco Express Forwarding or distributed Cisco Express Forwarding. To configure BGP, refer to the *BGP Configuration Guide*. To configure Cisco Express Forwarding and distributed Cisco Express Forwarding, refer to the *Cisco Express Forwarding Configuration Guide*.
- Define a policy.
- Apply the policy through BGP.
- Configure the BGP community list, BGP autonomous system path, or an access list and enable the policy on an interface.
- Enable CAR or WRED to use the policy. To enable CAR, see the chapter “Configuring Committed Access Rate” in the *Quality of Service Solutions Configuration Guide*. To configure WRED, see the

chapter “Configuring Weighted Random Early Detection” in the *Quality of Service Solutions Configuration Guide*.

Note Before the QoS Policy Propagation via BGP feature can work, you must enable BGP and Cisco Express Forwarding or distributed Cisco Express Forwarding on the router. Subinterfaces on ATM interfaces that have the **bgp-policy** command enabled must use Cisco Express Forwarding because distributed Cisco Express Forwarding is not supported. Distributed Cisco Express Forwarding uses the Versatile Interface Processor (VIP) rather than the Route Switch Processor (RSP) to perform forwarding functions.

Authentication Keys Management

Key management is a method of controlling authentication keys used by routing protocols. Not all protocols can use key management. Authentication keys are available for Director Response Protocol (DRP) Agent, EIGRP, and RIP Version 2.

Before you manage authentication keys, authentication must be enabled. See the appropriate protocol chapter to learn how to enable authentication for that protocol.

To manage authentication keys, see "Managing Authentication Keys".

How to Configure IP Routing Protocol-Independent Features

Redistributing Routing Information

You can redistribute routes from one routing domain into another, with or without controlling the redistribution with a route map. To control which routes are redistributed, configure a route map and reference the route map from the **redistribute** command.

The tasks in this section describe how to define the conditions for redistributing routes (a route map), how to redistribute routes, and how to remove options for redistributing routes, depending on the protocol being used.

Defining Conditions for Redistributing Routes

Route maps can be used to control route redistribution (or to implement policy-based routing). To define conditions for redistributing routes from one routing protocol into another, configure the **route-map** command. Then use at least one **match** command in route map configuration mode, as needed. At least one

match command would be used in this task because the purpose of the task is to illustrate how to define one or more conditions on which to base redistribution.

Note A route map is not required to have **match** commands; it is possible for it to have only **set** command(s). If there are no **match** commands, everything matches the route map.

Note There are many more **match** commands not shown in this task table. For additional **match** commands, see the *Cisco IOS Master Command List*.

Command or Action	Purpose
match as-path *path-list-number*	Matches a BGP autonomous system path access list.
match community {*standard-list-number* | *expanded-list-number* | *community-list-name* **match community**[**exact**]}	Matches a BGP community.
match ip address {*access-list-number* [*access-list-number...* | *access-list-name...*] | *access-list-name* [*access-list-number...*| *access-list-name*] | **prefix-list** *prefix-list-name* [*prefix-list-name...*]}	Matches routes that have a destination network address that is permitted to policy route packets or is permitted by a standard access list, an extended access list, or a prefix list.
match metric *metric-value*	Matches routes with the specified metric.
match ip next-hop {*access-list-number* | *access-list-name*} [*access-list-number* | *access-list-name*]	Matches a next-hop router address passed by one of the specified access lists.
match tag *tag-value* [*tag-value*]	Matches the specified tag value.
match interface *type number* [*type number*]	Matches routes that use the specified interface as the next hop.
match ip route-source {*access-list-number* | *access-list-name*} [*access-list-number* | *access-list-name*]	Matches the address specified by the advertised access lists.
match route-type {**local** | **internal** | **external** [**type-1** | **type-2**] | **level-1** | **level-2**}	Matches the specified route type.

To optionally specify the routing actions for the system to perform if the match criteria are met (for routes that are being redistributed by the route map), use one or more **set** commands in route map configuration mode, as needed.

Note A route map is not required to have **set** commands; it is possible for it to have only **match** command(s).

Note There are more **set** commands not shown in this task table. For additional **set** commands, see the *Cisco IOS Master Command List*.

Command or Action	Purpose
set community {*community-number* [**additive**] [**well-known**] | **none**}	Sets the community attribute (for BGP).
set dampening *halflife reuse suppress max-suppress-time*	Sets route dampening parameters (for BGP).
set local-preference *number-value*	Assigns a local preference value to a path (for BGP).
set origin {**igp** | **egp** *as-number* | **incomplete**}	Sets the route origin code.
set as-path{**tag** | **prepend** *as-path-string* }	Modifies the autonomous system path (for BGP).
set next-hop *next-hop*	Specifies the address of the next hop.
set automatic-tag	Enables automatic computation of the tag table.
set level {**level-1** | **level-2** | **level-1-2** | **stub-area** | **backbone**}	Specifies the areas to import routes.
set metric *metric-value*	Sets the metric value for redistributed routes (for any protocol, except EIGRP).
set metric *bandwidth delay reliability load mtu*	Sets the metric value for redistributed routes (for EIGRP only).
set metric-type {**internal** | **external** | **type-1** | **type-2**}	Sets the metric type for redistributed routes.
set metric-type internal	Sets the Multi Exit Discriminator (MED) value on prefixes advertised to the external BGP neighbor to match the Interior Gateway Protocol (IGP) metric of the next hop.
set tag *tag-value*	Sets a tag value to be applied to redistributed routes.

Redistributing Routes from One Routing Domain to Another

Perform this task to redistribute routes from one routing domain into another and to control route redistribution. This task shows how to redistribute OSPF routes into a BGP domain.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **router bgp** *autonomous-system*
4. **redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**match** {**internal** | **external** *type-value*}] [**tag** *tag-value*] [**route-map** *map-tag*] [**subnets**]
5. **default-metric** *number*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**router bgp** *autonomous-system* **Example:** `Router(config)# router bgp 109`	Enables a BGP routing process and enters router configuration mode.
Step 4	**redistribute** *protocol* [*process-id*] {**level-1** | **level-1-2** | **level-2**} [**metric** *metric-value*] [**metric-type** *type-value*] [**match** {**internal** | **external** *type-value*}] [**tag** *tag-value*] [**route-map** *map-tag*] [**subnets**] **Example:** `Router(config-router)# redistribute ospf 2 level-1`	Redistributes routes from the specified routing domain into another routing domain.

	Command or Action	Purpose
Step 5	**default-metric** *number* **Example:** `Router(config-router)# default-metric 10`	Sets the default metric value for redistributed routes. (BGP, OSPF, RIP). **Note** The metric value specified in the **redistribute** command supersedes the metric value specified using the **default-metric** command.
Step 6	**end** **Example:** `Router(config-router)# end`	Exits router configuration mode and returns to privileged EXEC mode.

Removing Options for Redistribution Routes

Caution Removing options that you have configured for the **redistribute** command requires careful use of the **no** form of the **redistribute** command to ensure that you obtain the result that you are expecting.

It is important to understand that different protocols implement the **no** version of the **redistribute** command differently:

- In BGP, OSPF, and RIP configurations, the **no redistribute** command removes only the specified keywords from the **redistribute** commands in the running configuration. They use the *subtractive keyword* method when redistributing from other protocols. For example, in the case of BGP, if you configure **no redistribute static route-map interior**, only the route map is removed from the redistribution, leaving **redistribute static** in place with no filter.
- The **no redistribute isis** command removes the IS-IS redistribution from the running configuration. IS-IS removes the entire command, regardless of whether IS-IS is the redistributed or redistributing protocol.
- EIGRP used the subtractive keyword method prior to EIGRP component version rel5. Starting with EIGRP component version rel5, the **no redistribute** command removes the entire **redistribute** command when redistributing from any other protocol.
- For **no redistribute connected**, the behavior is subtractive if the **redistribute** command is configured under **router bgp** or **router ospf**. The behavior is complete removal of the command if it is configured under **router isis** or **router eigrp**.

The following OSPF examples illustrate how various options are removed from the redistribution in router configuration mode.

Command or Action	Purpose
no redistribute connected metric 1000 subnets	Removes the configured metric-value of 1000 and the configured subnets and retains the **redistribute connected** command in the configuration.

Command or Action	Purpose
no redistribute connected metric 1000	Removes the configured metric-value of 1000 and retains the **redistribute connected subnets** command in the configuration.
no redistribute connected subnets	Removes the configured subnets and retains the **redistribute connected metric** *metric-value* command in the configuration.
no redistribute connected	Removes the **redistribute connected** command and any of the options that were configured for the command.

Configuring Routing Information Filtering

To filter routing protocol information, perform the tasks in the following sections. The tasks in the first section are required; the remaining sections are optional:

Note When routes are redistributed between OSPF processes, no OSPF metrics are preserved

Preventing Routing Updates Through an Interface

To prevent other routers on a local network from learning about routes dynamically, you can keep routing update messages from being sent through a router interface. Keeping routing update messages from being sent through a router interface prevents other systems on the interface from learning about routes dynamically. This feature applies to all IP-based routing protocols except BGP.

OSPF and IS-IS behave somewhat differently. In OSPF, the interface address that you specify as passive appears as a stub network in the OSPF domain. OSPF routing information is neither sent nor received through the specified router interface. In IS-IS, the specified IP addresses are advertised without actually running IS-IS on those interfaces.

To prevent routing updates through a specified interface, use the **passive-interface** *interface-type interface-number* command in router configuration mode. See the GUID-2D5A471C-3407-4179-AB3D-29846E96A786 for examples of configuring passive interfaces.

Configuring Default Passive Interfaces

To set all interfaces as passive by default and then activate only those interfaces that must have adjacencies set, perform the following task.

SUMMARY STEPS

1. Router> **enable**
2. Router# **configure terminal**
3. Router(config)# **router** *protocol*
4. Router(config-router)# **passive-interface default**
5. Router(config-router)# **no passive-interface** *interface-type*
6. Router(config-router)# **network** *network-address*[*options*]
7. Router(config-router)# **end**
8. Router# **show ip ospf interface**
9. Router# **show ip interface**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router> **enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Router# **configure terminal**	Enters global configuration mode.
Step 3	Router(config)# **router** *protocol*	Configures the routing protocol on the network.
Step 4	Router(config-router)# **passive-interface default**	Sets all interfaces as passive by default.
Step 5	Router(config-router)# **no passive-interface** *interface-type*	Activates only those interfaces that must have adjacencies set.
Step 6	Router(config-router)# **network** *network-address*[*options*]	Specifies the list of networks for the routing process. The *network-address* argument is an IP address written in dotted decimal notation--172.24.101.14, for example.
Step 7	Router(config-router)# **end**	Exits router configuration mode and returns to privileged EXEC mode.
Step 8	Router# **show ip ospf interface**	Displays interface information related to Open Shortest Path First (OSPF).
Step 9	Router# **show ip interface**	Displays the usability status of interfaces configured for IP.

See the section "Example Default Passive Interface" for an example of a default passive interface.

To verify that interfaces on your network have been set to passive, you could enter a network monitoring command such as the **show ip ospf interface** command, or you could verify the interfaces that you enabled as active using a command such as the **show ip interface** command.

Controlling the Advertising of Routes in Routing Updates

To prevent other routers from learning one or more routes, you can suppress routes from being advertised in routing updates. Suppressing routes in route updates prevents other routers from learning the interpretation of a particular device of one or more routes. You cannot specify an interface name in OSPF. When used for OSPF, this feature applies only to external routes.

To suppress routes from being advertised in routing updates, use the **distribute-list** {*access-list-number* | *access-list-name*} **out**[*interface-name* | *routing-process* | *as-number*] command in router configuration mode.

Controlling the Processing of Routing Updates

You might want to avoid processing certain routes listed in incoming updates. This feature does not apply to OSPF or IS-IS. To suppress routes in incoming updates, use the **distribute-list** {*access-list-number* | *access-list-name*} **in**[*interface-type interface-number*] command in router configuration mode.

Filtering Sources of Routing Information

To filter sources of routing information, use the **distance** *ip-address wildcard- mask*[*ip-standard-acl* | *ip-extended-acl* | *access-list-name*] command in router configuration mode.

Configuring precedence for policy-based routed packets

To configure the precedence and specify where the packets that pass the match criteria are output, perform the following task.

Note The **set ip next-hop** and **set ip default next-hop** commands are similar but have a different order of operation. Configuring the **set ip next-hop** command causes the system to use policy routing first and then use the routing table. Configuring the **set ip default next-hop** causes the system to use the routing table first and then policy-route the specified next hop.

SUMMARY STEPS

1. Router(config-route-map)# **set ip precedence**{*number* | *name*}
2. Router(config-route-map)# **set ip next-hop** *ip-address* [*ip-address*]
3. Router(config-route-map)# **set interface** *interface-type interface-number*[*... interface-type interface-number*]
4. Router(config-route-map)# **set ip default next-hop** *ip-address* [*ip-address*]
5. Router(config-route-map)# **set default interface** *interface-type interface-number*[*... interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-route-map)# **set ip precedence**{*number* | *name*}	Sets the precedence value in the IP header.
Step 2	Router(config-route-map)# **set ip next-hop** *ip-address* [*ip-address*]	Specifies the next hop to which to route the packet. **Note** The next hop must be an adjacent router.
Step 3	Router(config-route-map)# **set interface** *interface-type interface-number*[*... interface-type interface-number*]	Specifies the output interface for the packet.

	Command or Action	Purpose
Step 4	Router(config-route-map)# **set ip default next-hop** *ip-address* [*ip-address*]	Specifies the next hop to which to route the packet, if there is no explicit route for this destination. **Note** Like the **set ip next-hop**command, the **set ip default next-hop** command must specify an adjacent router.
Step 5	Router(config-route-map)# **set default interface** *interface-type interface-number*[... *interface-type interface-number*]	Specifies the output interface for the packet if there is no explicit route for the destination.

Configuring QoS Policy Propagation via BGP

The QoS Policy Propagation via BGP feature allows you to classify packets by IP precedence based on BGP community lists, BGP autonomous system paths, and access lists. After a packet has been classified, you can use other QoS features such as committed access rate (CAR) and Weighted Random Early Detection (WRED) to specify and enforce policies to fit your business model.

To configure Policy Propagation via BGP, perform the following basic tasks:

- Configure BGP and Cisco Express Forwarding (CEF) or distributed CEF (dCEF). To configure BGP, refer to the *Cisco IOS IP Routing: BGP Configuration Guide* . To configure CEF and dCEF, refer to the *Cisco IOS IP Switching Configuration Guide* .
- Define the policy.
- Apply the policy through BGP.
- Configure the BGP community list, BGP autonomous system path, or access list and enable the policy on an interface. For information about these tasks, see the tasks below.
- Enable CAR or WRED to use the policy. To enable CAR, see the chapter "Configuring Committed Access Rate" in the *Cisco IOS Quality of Service Solutions Configuration Guide*. To configure WRED, see the chapter "Configuring Weighted Random Early Detection" in the *Cisco IOS Quality of Service Solutions Configuration Guide*.

This section describes how to configure QoS Policy Propagation based on BGP community list, BGP autonomous system path, or access list. It assumes you have already configured BGP and CEF or dCEF.

To configure QoS Policy Propagation via BGP, perform the tasks described in the following sections. The tasks in the first three sections are required; the task in the remaining section is optional.

Note For the QoS Policy Propagation via BGP feature to work, you must enable BGP and CEF/dCEF on the router. Subinterfaces on an ATM interface that have the **bgp-policy**command enabled must use CEF mode because dCEF is not supported. dCEF uses the Versatile Interface Processor (VIP) rather than the Route Switch Processor (RSP) to perform forwarding functions.

For configuration examples, see "Examples QoS Policy Propagation via BGP Configuration."

- Configuring QoS Policy Propagation Based on Community Lists, page 22
- Configuring QoS Policy Propagation Based on the Autonomous System Path Attribute, page 23
- Configuring QoS Policy Propagation Based on an Access List, page 24
- Monitoring QoS Policy Propagation via BGP, page 26

Configuring QoS Policy Propagation Based on Community Lists

This section describes how to configure Policy Propagation via BGP using community lists. The tasks listed in this section are required unless noted as optional. This section assumes that you have already configured CEF/dCEF and BGP on your router.

Perform the following task to configure the router to propagate the IP precedence based on the community lists.

SUMMARY STEPS

1. Router> **enable**
2. Router# **configure terminal**
3. Router(config)# **route-map** *route-map-name* [**permit** | **deny**[*sequence-number*]]
4. Router(config-route-map)# **match community-list** *community-list-number* [**exact**]
5. Router(config-route-map)# **set ip precedence**[*number* | *name*]
6. Router(config-route-map)# **exit**
7. Router(config)# **router bgp** *autonomous-system*
8. Router(config-router)# **table-map** *route-map-name*
9. Router(config-router)# **exit**
10. Router(config)# **ip community-list** *community-list-number*{**permit** | **deny**} *community-number*
11. Router(config)# **interface** *interface-type interface-number*
12. Router(config-if)# **bgp-policy** {**source** | **destination**} **ip-prec-map**
13. Router(config-if)# **ip bgp-community new-format**
14. Router(config-if)# **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router> **enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Router# **configure terminal**	Enters global configuration mode.
Step 3	Router(config)# **route-map** *route-map-name* [**permit** | **deny**[*sequence-number*]]	Defines a route map to control redistribution and enters route map configuration mode.
Step 4	Router(config-route-map)# **match community-list** *community-list-number* [**exact**]	Matches a BGP community list.
Step 5	Router(config-route-map)# **set ip precedence**[*number* | *name*]	Sets the IP Precedence field when the community list matches. You can specify either a precedence number or name.
Step 6	Router(config-route-map)# **exit**	Exits route map configuration mode and returns the router to global configuration mode.
Step 7	Router(config)# **router bgp** *autonomous-system*	Enters router configuration mode.

	Command or Action	Purpose
Step 8	Router(config-router)# **table-map** *route-map-name*	Modifies the metric and tag values when the IP routing table is updated with BGP learned routes.
Step 9	Router(config-router)# **exit**	Exits router configuration mode and returns the router to global configuration mode.
Step 10	Router(config)# **ip community-list** *community-list-number*{**permit** | **deny**} *community-number*	Creates a community list for BGP and controls access to it.
Step 11	Router(config)# **interface** *interface-type interface-number*	Specifies the interfaces (or subinterface) and enters interface configuration mode.
Step 12	Router(config-if)# **bgp-policy** {**source** | **destination**} **ip-prec-map**	Classifies packets using IP Precedence.
Step 13	Router(config-if)# **ip bgp-community new-format**	(Optional) Configures a new community format so that the community number is displayed in the short form.
Step 14	Router(config-if)# **end**	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring QoS Policy Propagation Based on the Autonomous System Path Attribute

This section describes how to configure QoS Policy Propagation via BGP based on the autonomous system path. This section assumes that you have already configured CEF/dCEF and BGP on your router.

Perform the following task to configure the router to propagate the IP precedence based on the autonomous system path attribute.

SUMMARY STEPS

1. Router> **enable**
2. Router# **configure terminal**
3. Router(config)# **route-map** *route-map-name* [**permit** | **deny**[*sequence-number*]]
4. Router(config-route-map)# **match as-path** *path-list-number*
5. Router(config-route-map)# **set ip precedence**[*number* | *name*]
6. Router(config-route-map)# **exit**
7. Router(config)# **router bgp** *autonomous-system*
8. Router(config-router)# **table-map** *route-map-name*
9. Router(config-router)# **exit**
10. Router(config)# **ip as-path access-list** *access-list-number*{**permit** | **deny**} *as-regular-expression*
11. Router(config)# **interface** *interface-type interface-number*
12. Router(config-if)# **bgp-policy** {**source** | **destination**} **ip-prec-map**
13. Router(config-if)# **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router> **enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Router# **configure terminal**	Enters global configuration mode.
Step 3	Router(config)# **route-map** *route-map-name* [**permit** | **deny**[*sequence-number*]]	Defines a route map to control redistribution and enters route-map configuration mode.
Step 4	Router(config-route-map)# **match as-path** *path-list-number*	Matches a BGP autonomous system path access list.
Step 5	Router(config-route-map)# **set ip precedence**[*number* | *name*]	Sets the IP Precedence field when the autonomous system path matches. Specifies either a precedence number or name.
Step 6	Router(config-route-map)# **exit**	Exits route map configuration mode and returns the router to global configuration mode.
Step 7	Router(config)# **router bgp** *autonomous-system*	Enters router configuration mode.
Step 8	Router(config-router)# **table-map** *route-map-name*	Modifies the metric and tag values when the IP routing table is updated with BGP learned routes.
Step 9	Router(config-router)# **exit**	Exits router configuration mode and returns the router to global configuration mode.
Step 10	Router(config)# **ip as-path access-list** *access-list-number*{**permit** | **deny**} *as-regular-expression*	Defines an autonomous system path access list.
Step 11	Router(config)# **interface** *interface-type interface-number*	Specifies the interfaces (or subinterface) and enters interface configuration mode.
Step 12	Router(config-if)# **bgp-policy** {**source** | **destination**} **ip-prec-map**	Classifies packets using IP Precedence.
Step 13	Router(config-if)# **end**	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring QoS Policy Propagation Based on an Access List

This section describes how to configure QoS Policy Propagation via BGP based on an access list. This section assumes you have already configured CEF/dCEF and BGP on your router.

To configure the router to propagate the IP Precedence based on an access list, perform the following task.s

SUMMARY STEPS

1. Router> **enable**
2. Router# **configure terminal**
3. Router(config)# **route-map** *route-map-name* [**permit** | **deny**[*sequence-number*]]
4. Router(config-route-map)# **match ip address** *access-list-number*
5. Router(config-route-map)# **set ip precedence**[*number* | *name*]
6. Router(config-route-map)# **exit**
7. Router(config)# **router bgp** *autonomous-system*
8. Router(config-router)# **table-map** *route-map-name*
9. Router(config-router)# **exit**
10. Router(config)# **access-list** *access-list-number* {**permit** | **deny**} *source*
11. Router(config)# **interface** *interface-type interface-number*
12. Router(config-if)# **bgp-policy** {**source** | **destination**} **ip-prec-map**
13. Router(config-if)# **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router> **enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Router# **configure terminal**	Enters global configuration mode.
Step 3	Router(config)# **route-map** *route-map-name* [**permit** | **deny**[*sequence-number*]]	Defines a route map to control redistribution and enters route map configuration mode.
Step 4	Router(config-route-map)# **match ip address** *access-list-number*	Matches an access list.
Step 5	Router(config-route-map)# **set ip precedence**[*number* | *name*]	Sets the IP Precedence field when the autonomous system path matches.
Step 6	Router(config-route-map)# **exit**	Exits route map configuration mode and returns the router to global configuration mode.
Step 7	Router(config)# **router bgp** *autonomous-system*	Enters router configuration mode.
Step 8	Router(config-router)# **table-map** *route-map-name*	Modifies the metric and tag values when the IP routing table is updated with BGP learned routes.
Step 9	Router(config-router)# **exit**	Exits router configuration mode and returns the router to global configuration mode.
Step 10	Router(config)# **access-list** *access-list-number* {**permit** | **deny**} *source*	Defines an access list.
Step 11	Router(config)# **interface** *interface-type interface-number*	Specifies the interfaces (or subinterface) and enters interface configuration mode.

	Command or Action	Purpose
Step 12	Router(config-if)# **bgp-policy** {**source** | **destination**} **ip-prec-map**	Classifies packets using IP Precedence.
Step 13	Router(config-if)# **end**	Exits interface configuration mode and returns to privileged EXEC mode.

Monitoring QoS Policy Propagation via BGP

To monitor the QoS Policy Propagation via BGP configuration, use the following commands in EXEC mode, as needed. The commands listed in this section are optional.

Command or Action	Purpose
`Router#` **show ip bgp**	Displays entries in the BGP routing table, to verify that the correct community is set on the prefixes.
`Router#` **show ip bgp community-list** *community-list-number*	Displays routes permitted by the BGP community list, to verify that the correct prefixes are selected.
`Router#` **show ip cef** *network*	Displays entries in the Forwarding Information Base (FIB) table based on the IP address, to verify that CEF has the correct precedence value for the prefix.
`Router#` **show ip interface**	Displays information about the interface.
`Router#` **show ip route** *prefix*	Displays the current status of the routing table, to verify that the correct precedence values are set on the prefixes.

Managing Authentication Keys

To manage authentication keys, define a key chain, identify the keys that belong to the key chain, and specify how long each key is valid. Each key has its own key identifier (specified with the **key**key-chain configuration command), which is stored locally. The combination of the key identifier and the interface associated with the message uniquely identifies the authentication algorithm and Message Digest 5 (MD5) authentication key in use.

You can configure multiple keys with lifetimes. Only one authentication packet is sent, regardless of how many valid keys exist. The software examines the key numbers in order from lowest to highest and uses the first valid key that it encounters. The lifetimes allow for overlap during key changes. Note that the router must know the time. Refer to the Network Time Protocol (NTP) and calendar commands in the "Performing Basic System Management" chapter of the *Network Management Configuration Guide*.

To manage authentication keys, perform the following task.

SUMMARY STEPS

1. Router> **enable**
2. Router# **configure terminal**
3. Router(config)# **key chain** *name-of-chain*
4. Router(config-keychain)# **key** *number*
5. Router(config-keychain-key)# **key-string** *text*
6. Router(config-keychain-key)# **accept-lifetime** *start-time*{**infinite** | *end-time*| **duration** *seconds*}
7. Router(config-keychain-key)# **send-lifetime** *start-time*{**infinite** | *end-time* | **duration** *seconds*}
8. Router(config-keychain-key)# **end**
9. Router# **show key chain**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router> **enable**	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	Router# **configure terminal**	Enters global configuration mode.
Step 3	Router(config)# **key chain** *name-of-chain*	Identifies a key chain.
Step 4	Router(config-keychain)# **key** *number*	Identifies the key number in key chain configuration mode.
Step 5	Router(config-keychain-key)# **key-string** *text*	Identifies the key string in key chain configuration mode.
Step 6	Router(config-keychain-key)# **accept-lifetime** *start-time*{**infinite** | *end-time*| **duration** *seconds*}	Specifies the time period during which the key can be received.
Step 7	Router(config-keychain-key)# **send-lifetime** *start-time*{**infinite** | *end-time* | **duration** *seconds*}	Specifies the time period during which the key can be sent.
Step 8	Router(config-keychain-key)# **end**	Exits key chain key configuration mode and returns to privileged EXEC mode.
Step 9	Router# **show key chain**	Displays authentication key information.

For examples of key management, see the Examples Key Management, page 40.

Monitoring and Maintaining the IP Network

You can remove all contents of a particular cache, table, or database. You also can display specific statistics. The following sections describe each of these tasks.

Clearing Routes from the IP Routing Table

You can remove all contents of a particular table. Clearing a table can become necessary when the contents of the particular structure have become, or are suspected to be, invalid.

To clear one or more routes from the IP routing table, use the **clear ip route** {*network* [*mask*] | *****} command in EXEC mode.

Displaying System and Network Statistics

You can display specific statistics such as the contents of IP routing tables, caches, and databases. Information provided can be used to determine resource utilization and solve network problems. You can also display information about node reachability and discover the routing path that packets leaving your device are taking through the network.

Command or Action	Purpose
`Router#` **show ip cache policy**	Displays the cache entries in the policy route cache.
`Router#` **show ip local policy**	Displays the local policy route map if one exists.
`Router#` **show ip policy**	Displays policy route maps.
`Router#` **show ip protocols**	Displays the parameters and current state of the active routing protocol process.
`Router#` **show ip route** [*ip-address* [*mask*] [**longer-prefixes**] | *protocol* [*process-id*] | **list** {*access-list-number* | *access-list-name*} | **static download**]	Displays the current state of the routing table.
`Router#` **show ip route summary**	Displays the current state of the routing table in summary form.
`Router#` **show ip route supernets-only**	Displays supernets.
`Router#` **show key chain** [*name-of-chain*]	Displays authentication key information.
`Router#` **show route-map** [*map-name*]	Displays all route maps configured or only the one specified.

Configuration Examples for Configuring IP Routing Protocol-Independent Features

Example: Configuring Redistribution Routes

Example Static Routing Redistribution

In the example that follows, three static routes are specified, two of which are to be advertised. The static routes are created by specifying the **redistribute static** router configuration command and then specifying an access list that allows only those two networks to be passed to the EIGRP process. Any redistributed static routes should be sourced by a single router to minimize the likelihood of creating a routing loop.

```
Router(config)# ip route 192.168.2.0 255.255.255.0 192.168.7.65
Router(config)# ip route 192.168.5.0 255.255.255.0 192.168.7.65
Router(config)# ip route 10.10.10.0 255.255.255.0 10.20.1.2
Router(config)# !
Router(config)# access-list 3 permit 192.168.2.0 0.0.255.255
Router(config)# access-list 3 permit 192.168.5.0 0.0.255.255
Router(config)# access-list 3 permit 10.10.10.0 0.0.0.255
Router(config)# !
Router(config)# router eigrp 1
Router(config-router)# network 192.168.0.0
Router(config-router)# network 10.10.10.0
Router(config-router)# redistribute static metric 10000 100 255 1 1500
Router(config-router)# distribute-list 3 out static
```

Example: EIGRP Redistribution

Each EIGRP routing process provides routing information to only one autonomous system. The Cisco IOS software must run a separate EIGRP process and maintain a separate routing database for each autonomous system that the software services. However, you can transfer routing information among routing databases.

In the following example, network 10.0.0.0 is configured under EIGRP autonomous system 1 and network 192.168.7.0 is configured under EIGRP autonomous system 101:

```
Router(config)# router eigrp 1
Router(config-router)# network 10.0.0.0
Router(config-router)# exit
Router(config)# router eigrp 101
Router(config-router)# network 192.168.7.0
```

In the following example, routes from the 192.168.7.0 network are redistributed into autonomous system 1 (without passing any other routing information from autonomous system 101):

```
Router(config)# access-list 3 permit 192.168.7.0
Router(config)# !
Router(config)# route-map 101-to-1 permit 10
Router(config-route-map)# match ip address 3
Router(config-route-map)# set metric 10000 100 1 255 1500
Router(config-route-map)# exit
Router(config)# router eigrp 1
Router(config-router)# redistribute eigrp 101 route-map 101-to-1
Router(config-router)#!
```

The following example is an alternative way to redistribute routes from the 192.168.7.0 network into autonomous system 1.This method does not allow you to set the metric for redistributed routes.

```
Router(config)# access-list 3 permit 192.168.7.0
Router(config)# !
Router(config)# router eigrp 1
Router(config-router)# redistribute eigrp 101
Router(config-router)# distribute-list 3 out eigrp 101
Router(config-router)# !
```

Example: Mutual Redistribution Between EIGRP and RIP

Consider a WAN at a university that uses RIP as an interior routing protocol. Assume that the university wants to connect its WAN to regional network 172.16.0.0, which uses EIGRP as the routing protocol. The goal in this case is to advertise the networks in the university network to routers in the regional network.

Mutual redistribution is configured between EIGRP and RIP in the following example:

```
Router(config)# access-list 10 permit 172.16.0.0
Router(config)# !
Router(config)# router eigrp 1
Router(config-router)# network 172.16.0.0
Router(config-router)# redistribute rip metric 10000 100 255 1 1500
Router(config-router)# distribute-list 10 out rip
Router(config-router)# exit
Router(config)# router rip
Router(config-router)# redistribute eigrp 1
Router(config-router)# !
```

In this example, an EIGRP routing process is started. The **network** router configuration command specifies that network 172.16.0.0 (the regional network) is to send and receive EIGRP routing information. The **redistribute** router configuration command specifies that RIP-derived routing information be advertised in routing updates. The **default-metric** router configuration command assigns an EIGRP metric to all RIP-derived routes. The **distribute-list** router configuration command instructs the Cisco software to use access list 10 (not defined in this example) to limit the entries in each outgoing update. The access list prevents unauthorized advertising of university routes to the regional network.

Example: Mutual Redistribution Between EIGRP and BGP

In the following example, mutual redistribution is configured between EIGRP and BGP.

Routes from EIGRP routing process 101 are injected into BGP autonomous system 50000. A filter is configured to ensure that the correct routes are advertised, in this case, three networks. Routes from BGP autonomous system 50000 are injected into EIGRP routing process 101. The same filter is used.

```
Router(config)# ! All networks that should be advertised from R1 are controlled with
ACLs:
Router(config)# access-list 1 permit 172.18.0.0 0.0.255.255
Router(config)# access-list 1 permit 172.16.0.0 0.0.255.255
Router(config)# access-list 1 permit 172.25.0.0 0.0.255.255
Router(config)# ! Configuration for router R1:
Router(config)# router bgp 50000
Router(config-router)# network 172.18.0.0
Router(config-router)# network 172.16.0.0
Router(config-router)# neighbor 192.168.10.1 remote-as 2
Router(config-router)# neighbor 192.168.10.15 remote-as 1
Router(config-router)# neighbor 192.168.10.24 remote-as 3
Router(config-router)# redistribute eigrp 101
Router(config-router)# distribute-list 1 out eigrp 101
Router(config-router)# exit
Router(config)# router eigrp 101
Router(config-router)# network 172.25.0.0
Router(config-router)# redistribute bgp 50000
```

```
Router(config-router)# distribute-list 1 out bgp 50000
Router(config-router)# !
```

Caution BGP should be redistributed into an IGP when there are no other suitable options. Redistribution from BGP into any IGP should be applied with proper filtering by using distribute-lists, IP prefix lists, and route map statements to limit the number of prefixes.

Example: OSPF Routing and Route Redistribution

OSPF typically requires coordination among many internal routers, ABRs, and Autonomous System Boundary Routers (ASBRs). At a minimum, OSPF-based routers can be configured with all default parameter values, with no authentication, and with interfaces assigned to areas.

This section provides the following configuration examples:

- The first example shows simple configurations illustrating basic OSPF commands.
- The second example shows configurations for an internal router, ABR, and ASBR within a single, arbitrarily assigned OSPF autonomous system.
- The third example illustrates a more complex configuration and the application of various tools available for controlling OSPF-based routing environments.

- Examples Basic OSPF Configuration, page 31
- Example Internal Router ABR and ASBRs Configuration, page 32
- Example Complex OSPF Configuration, page 35

Examples Basic OSPF Configuration

The following example illustrates a simple OSPF configuration that enables OSPF routing process 1, attaches Ethernet interface 0 to area 0.0.0.0, and redistributes RIP into OSPF and OSPF into RIP:

```
Router(config)# interface Ethernet 0
Router(config-if)# ip address 172.16.1.1 255.255.255.0
Router(config-if)# ip ospf cost 1
Router(config-if)# exit
Router(config)# interface Ethernet 1
Router(config-if)# ip address 172.17.1.1 255.255.255.0
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# network 172.18.0.0 0.0.255.255 area 0.0.0.0
Router(config-router)# redistribute rip metric 1 subnets
Router(config-router)# exit
Router(config)# router rip
Router(config-router)# network 172.17.0.0
Router(config-router)# redistribute ospf 1
Router(config-router)# default-metric 1
Router(config-router)# !
```

The following example illustrates the assignment of four area IDs to four IP address ranges. In the example, OSPF routing process 1 is initialized, and four OSPF areas are defined: 10.9.50.0, 2, 3, and 0. Areas 10.9.50.0, 2, and 3 mask-specific address ranges, whereas area 0 enables OSPF for all other networks.

```
Router(config)# router ospf 1
Router(config-router)# network 172.18.20.0 0.0.0.255 area 10.9.50.0
Router(config-router)# network 172.18.0.0 0.0.255.255 area 2
Router(config-router)# network 172.19.10.0 0.0.0.255 area 3
Router(config-router)# network 0.0.0.0 255.255.255.255 area 0
Router(config-router)# exit
Router(config)# ! Ethernet interface 0 is in area 10.9.50.0:
```

```
Router(config)# interface Ethernet 0
Router(config-if)# ip address 172.18.20.5 255.255.255.0
Router(config-if)# exit
Router(config)# ! Ethernet interface 1 is in area 2:
Router(config)# interface Ethernet 1
Router(config-if)# ip address 172.18.1.5 255.255.255.0
Router(config-if)# exit
Router(config)# ! Ethernet interface 2 is in area 2:
Router(config)# interface Ethernet 2

Router(config-if)# ip address 172.18.2.5 255.255.255.0
Router(config-if)# exit
Router(config)# ! Ethernet interface 3 is in area 3:
Router(config)# interface Ethernet 3
Router(config-if)# ip address 172.19.10.5 255.255.255.0
Router(config-if)# exit
Router(config)# ! Ethernet interface 4 is in area 0:
Router(config)# interface Ethernet 4
Router(config-if)# ip address 172.19.1.1 255.255.255.0
Router(config-if)# exit
Router(config)# ! Ethernet interface 5 is in area 0:
Router(config)# interface Ethernet 5
Router(config-if)# ip address 10.1.0.1 255.255.0.0
Router(config-if)# !
```

Each **network** router configuration command is evaluated sequentially, so the specific order of these commands in the configuration is important. The Cisco IOS software sequentially evaluates the *address*/*wildcard-mask* pair for each interface. See the *Cisco IOS IP Routing: Protocol-Independent Command Reference* for more information.

Consider the first **network** command. Area ID 10.9.50.0 is configured for the interface on which subnet 172.18.20.0 is located. Assume that a match is determined for Ethernet interface 0. Ethernet interface 0 is attached to Area 10.9.50.0 only.

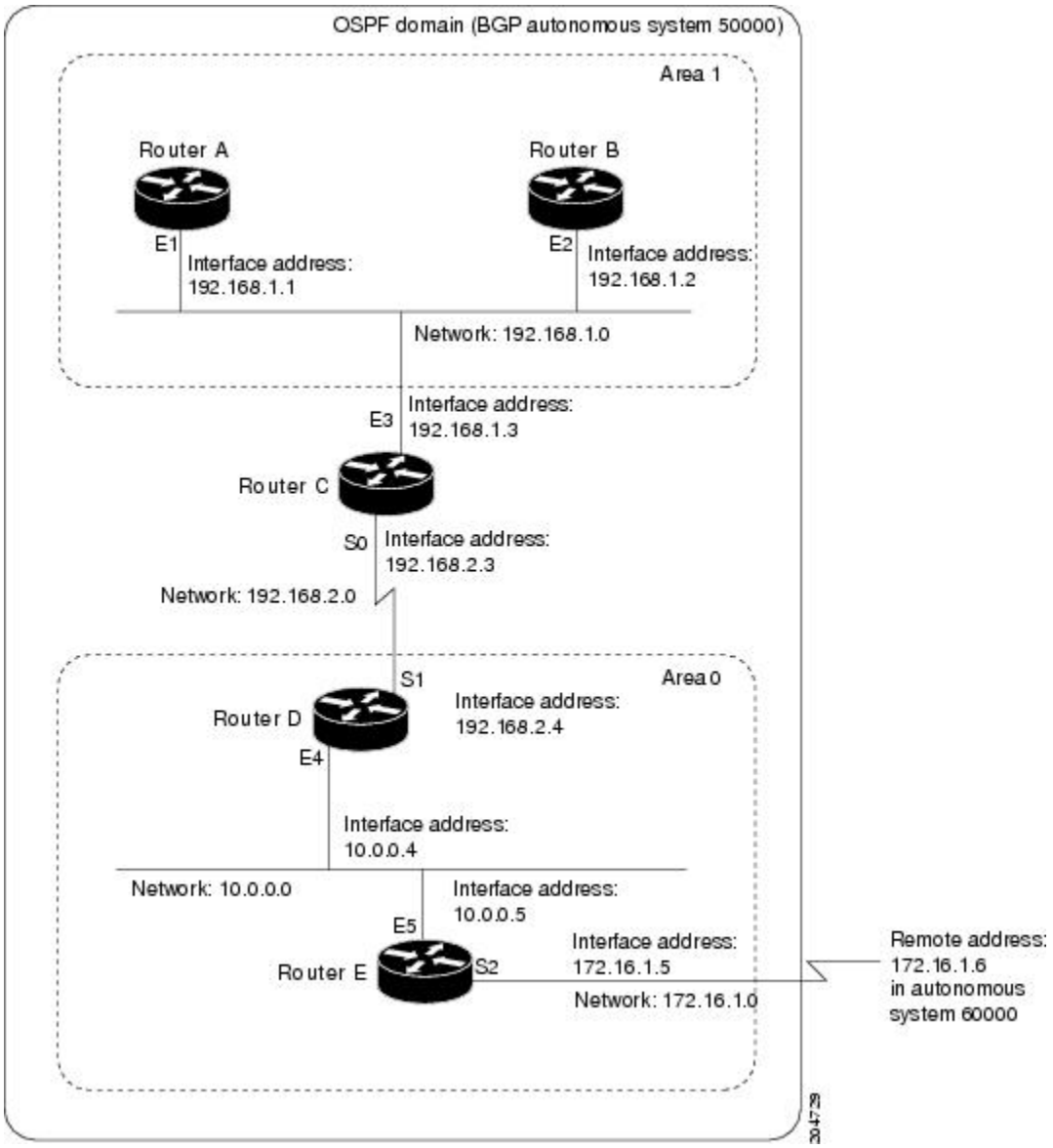
The second **network**command is evaluated next. For Area 2, the same process is then applied to all interfaces (except Ethernet interface 0). Assume that a match is determined for Ethernet interface 1. OSPF is then enabled for that interface, and Ethernet 1 is attached to Area 2.

This process of attaching interfaces to OSPF areas continues for all **network** commands. Note that the last **network** command in this example is a special case. With this command, all available interfaces (not explicitly attached to another area) are attached to Area 0.

Example Internal Router ABR and ASBRs Configuration

The figure below provides a general network map that illustrates a sample configuration for several routers within a single OSPF autonomous system.

Figure 1 Example OSPF Autonomous System Network Map



In this configuration, five routers are configured in OSPF autonomous system 1:

- Router A and Router Bare both internal routers within area 1.
- Router C is an OSPF ABR. Note that for Router C, area 1 is assigned to E3 and Area 0 is assigned to S0.
- Router D is an internal router in area 0 (backbone area). In this case, both **network** router configuration commands specify the same area (area 0, or the backbone area).
- Router E is an OSPF ASBR. Note that BGP routes are redistributed into OSPF and that these routes are advertised by OSPF.

Note It is not necessary to include definitions of all areas in an OSPF autonomous system in the configuration of all routers in the autonomous system. You must define only the *directly* connected areas. In the example that follows, routes in Area 0 are learned by the routers in area 1 (Router A and Router B) when the ABR (Router C) injects summary LSAs into area 1.

Autonomous system 60000 is connected to the outside world via the BGP link to the external peer at IP address 172.16.1.6.

Following is the example configuration for the general network map shown in the figure above.

Router A Configuration--Internal Router

```
Router(config)# interface Ethernet 1
Router(config-if)# ip address 192.168.1.1 255.255.255.0
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# network 192.168.1.0 0.0.0.255 area 1
Router(config-router)# exit
```

Router B Configuration--Internal Router

```
Router(config)# interface Ethernet 2
Router(config-if)# ip address 192.168.1.2 255.255.255.0
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# network 192.168.1.0 0.0.0.255 area 1
Router(config-router)# exit
```

Router C Configuration--ABR

```
Router(config)# interface Ethernet 3
Router(config-if)# ip address 192.168.1.3 255.255.255.0
Router(config-if)# exit
Router(config)# interface Serial 0
Router(config-if)# ip address 192.168.2.3 255.255.255.0
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# network 192.168.1.0 0.0.0.255 area 1
Router(config-router)# network 192.168.2.0 0.0.0.255 area 0
Router(config-router)# exit
```

Router D Configuration--Internal Router

```
Router(config)# interface Ethernet 4
Router(config-if)# ip address 10.0.0.4 255.0.0.0
Router(config-if)# exit
Router(config)# interface Serial 1
Router(config-if)# ip address 192.168.2.4 255.255.255.0
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# network 192.168.2.0 0.0.0.255 area 0
Router(config-router)# network 10.0.0.0 0.255.255.255 area 0
Router(config-router)# exit
```

Router E Configuration--ASBR

```
Router(config)# interface Ethernet 5
```

```
Router(config-if)# ip address 10.0.0.5 255.0.0.0
Router(config-if)# exit
Router(config)# interface Serial 2
Router(config-if)# ip address 172.16.1.5 255.255.255.0
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# network 10.0.0.0 0.255.255.255 area 0
Router(config-router)# redistribute bgp 50000 metric 1 metric-type 1
Router(config-router)# exit
Router(config)# router bgp 50000
Router(config-router)# network 192.168.0.0
Router(config-router)# network 10.0.0.0
Router(config-router)# neighbor 172.16.1.6 remote-as 60000
```

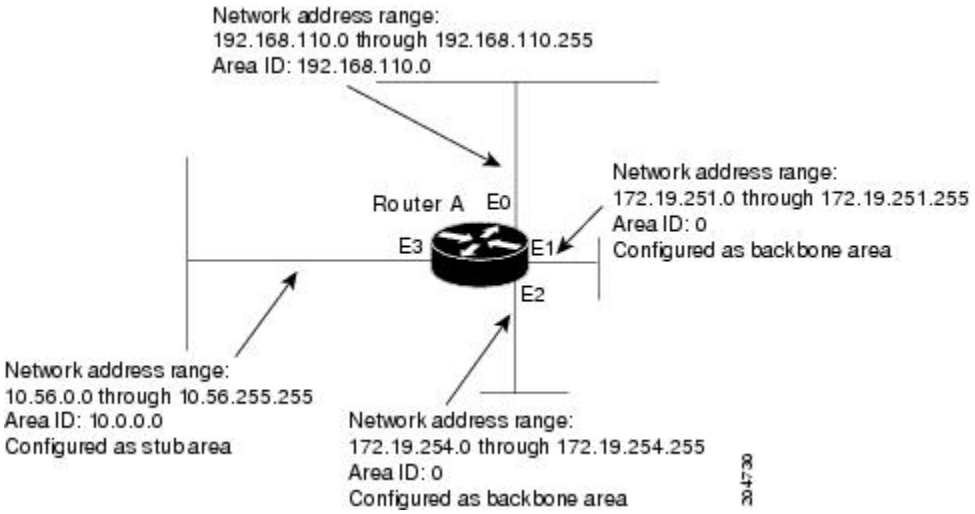
Example Complex OSPF Configuration

The following example configuration accomplishes several tasks in setting up an ABR. These tasks can be split into two general categories:

- Basic OSPF configuration
- Route redistribution

The specific tasks outlined in this configuration are detailed briefly in the following descriptions. The figure below illustrates the network address ranges and area assignments for the interfaces.

Figure 2 Interface and Area Specifications for OSPF Configuration Example



The basic configuration tasks in this example are as follows:

- Configure address ranges for Ethernet interface 0 through Ethernet interface 3.
- Enable OSPF on each interface.
- Set up an OSPF authentication password for each area and network.
- Assign link-state metrics and other OSPF interface configuration options.
- Create a *stub area* with area ID 10.0.0.0. (Note that the **authentication** and **stub** options of the **area** router configuration command are specified with separate **area** command entries, but they can be merged into a single **area** command.)
- Specify the backbone area (area 0).

Configuration tasks associated with redistribution are as follows:

- Redistribute EIGRP and RIP into OSPF with various options set (including **metric-type**, **metric**, **tag**, and **subnet**).
- Redistribute EIGRP and OSPF into RIP.

The following is an example OSPF configuration:

```
Router(config)# interface Ethernet 0
Router(config-if)# ip address 192.168.110.201 255.255.255.0
Router(config-if)# ip ospf authentication-key abcdefgh
Router(config-if)# ip ospf cost 10
Router(config-if)# exit
Router(config)# interface Ethernet 1
Router(config-if)# ip address 172.19.251.201 255.255.255.0
Router(config-if)# ip ospf authentication-key ijklmnop
Router(config-if)# ip ospf cost 20
Router(config-if)# ip ospf retransmit-interval 10
Router(config-if)# ip ospf transmit-delay 2
Router(config-if)# ip ospf priority 4
Router(config-if)# exit
Router(config)# interface Ethernet 2
Router(config-if)# ip address 172.19.254.201 255.255.255.0
Router(config-if)# ip ospf authentication-key abcdefgh
Router(config-if)# ip ospf cost 10
Router(config-if)# exit
Router(config)# interface Ethernet 3
Router(config-if)# ip address 10.56.0.201 255.255.0.0
Router(config-if)# ip ospf authentication-key ijklmnop
Router(config-if)# ip ospf cost 20
Router(config-if)# ip ospf dead-interval 80
Router(config-if)# exit
```

In the following configuration, OSPF is on network 172.19.0.0:

```
Router(config)# router ospf 1
Router(config-router)# network 10.0.0.0 0.255.255.255 area 10.0.0.0
Router(config-router)# network 192.168.110.0 0.0.0.255 area 192.68.110.0
Router(config-router)# network 172.19.0.0 0.0.255.255 area 0
Router(config-router)# area 0 authentication
Router(config-router)# area 10.0.0.0 stub
Router(config-router)# area 10.0.0.0 authentication
Router(config-router)# area 10.0.0.0 default-cost 20
Router(config-router)# area 192.168.110.0 authentication
Router(config-router)# area 10.0.0.0 range 10.0.0.0 255.0.0.0
Router(config-router)# area 192.168.110.0 range 192.168.110.0 255.255.255.0
Router(config-router)# area 0 range 172.19.251.0 255.255.255.0
Router(config-router)# area 0 range 172.19.254.0 255.255.255.0
Router(config-router)# redistribute eigrp 200 metric-type 2 metric 1 tag 200 subnets
Router(config-router)# redistribute rip metric-type 2 metric 1 tag 200
Router(config-router)# exit
```

In the following configuration, EIGRP autonomous system 1 is on 172.19.0.0:

```
Router(config)# router eigrp 1
Router(config-router)# network 172.19.0.0
Router(config-router)# exit
Router(config)# ! RIP for 192.168.110.0:
Router(config)# router rip
Router(config-router)# network 192.168.110.0
Router(config-router)# redistribute eigrp 1 metric 1
Router(config-router)# redistribute ospf 201 metric 1
Router(config-router)# exit
```

Example Default Metric Values Redistribution

The following example shows a router in autonomous system 1 that is configured to run both RIP and EIGRP. The example advertises EIGRP-derived routes using RIP and assigns the EIGRP-derived routes a RIP metric of 10.

```
Router(config)# router rip
Router(config-router)# default-metric 10
Router(config-router)# redistribute eigrp 1
Router(config-router)# exit
```

Example Default Passive Interface

The following example configures the network interfaces, sets all interfaces that are running OSPF as passive, and then enables serial interface 0:

```
Router(config)# interface Ethernet 0
Router(config-if)# ip address 172.19.64.38 255.255.255.0 secondary
Router(config-if)# ip address 172.19.232.70 255.255.255.240
Router(config-if)# no ip directed-broadcast
Router(config-if)# exit
Router(config)# interface Serial 0
Router(config-if)# ip address 172.24.101.14 255.255.255.252
Router(config-if)# no ip directed-broadcast
Router(config-if)# no ip mroute-cache
Router(config-if)# exit
Router(config)# interface TokenRing 0
Router(config-if)# ip address 172.20.10.4 255.255.255.0
Router(config-if)# no ip directed-broadcast
Router(config-if)# no ip mroute-cache
Router(config-if)# ring-speed 16
Router(config-if)# exit
Router(config)# router ospf 1
Router(config-router)# passive-interface default
Router(config-router)# no passive-interface Serial 0
Router(config-router)# network 172.16.10.0 0.0.0.255 area 0
Router(config-router)# network 172.19.232.0 0.0.0.255 area 4
Router(config-router)# network 172.24.101.0 0.0.0.255 area 4
Router(config-router)# exit
```

Example: Configuring an IP Default Gateway as a Static IP Next Hop When IP Routing Is Disabled

The following example shows how to configure IP address 172.16.5.4 as the default route when IP routing is disabled:

```
Router> enable
Router# configure terminal
Router(conf)# no ip routing
Router(conf)# ip default-gateway 172.16.15.4
```

Examples QoS Policy Propagation via BGP Configuration

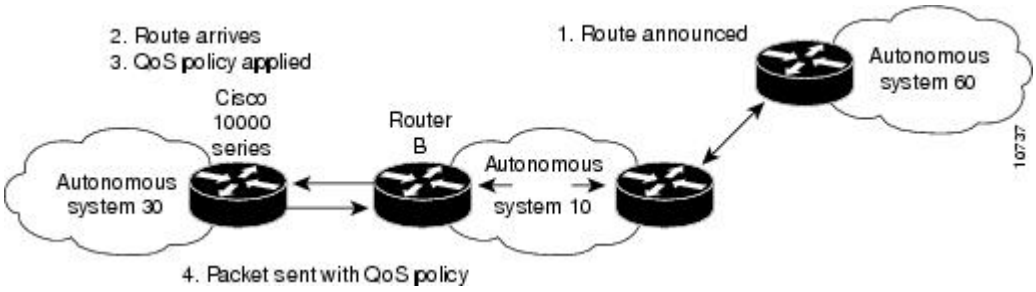
The following example shows how to create route maps to match access lists, BGP community lists, and BGP autonomous system paths, and apply IP precedence to routes learned from neighbors.

For information on how to configure QoS Policy Propagation via BGP, see the section Configuring QoS Policy Propagation via BGP, page 21 in this document.

In the figure below, Router A (Cisco 10000 Series) learns routes from autonomous system 10 and autonomous system 60. QoS policy is applied to all packets that match the defined route maps. Any packets

from Router A (Cisco 10000 Series) to autonomous system 10 or autonomous system 60 are sent the appropriate QoS policy, as the numbered steps indicate.

Figure 3 Router Learning Routes and Applying QoS Policy



Router A (Cisco 10000 Series) Configuration

```
interface serial 5/0/0/1:0
ip address 10.28.38.2 255.255.255.0
bgp-policy destination ip-prec-map
no ip mroute-cache
no cdp enable
frame-relay interface-dlci 20 IETF
router bgp 30
 table-map precedence-map
 neighbor 10.20.20.1 remote-as 10
 neighbor 10.20.20.1 send-community
!
ip bgp-community new-format
!
! Match community 1 and set the IP Precedence to priority
route-map precedence-map permit 10
 match community 1
 set ip precedence priority
!
! Match community 2 and set the IP Precedence to immediate
route-map precedence-map permit 20
 match community 2
 set ip precedence immediate
!
! Match community 3 and set the IP Precedence to flash
route-map precedence-map permit 30
 match community 3
 set ip precedence flash
!
! Match community 4 and set the IP Precedence to flash-override
route-map precedence-map permit 40
 match community 4
 set ip precedence flash-override
!
! Match community 5 and set the IP Precedence to critical
route-map precedence-map permit 50
 match community 5
 set ip precedence critical
!
! Match community 6 and set the IP Precedence to internet
route-map precedence-map permit 60
 match community 6
 set ip precedence internet
!
! Match community 7 and set the IP Precedence to network
route-map precedence-map permit 70
 match community 7
 set ip precedence network
!
! Match ip address access list 69 or match AS path 1
```

```
! and set the IP Precedence to critical
route-map precedence-map permit 75
 match ip address 69
 match as-path 1
 set ip precedence critical
!
! For everything else, set the IP Precedence to routine
route-map precedence-map permit 80
 set ip precedence routine
!
! Define the community lists
ip community-list 1 permit 60:1
ip community-list 2 permit 60:2
ip community-list 3 permit 60:3
ip community-list 4 permit 60:4
ip community-list 5 permit 60:5
ip community-list 6 permit 60:6
ip community-list 7 permit 60:7
!
! Define the AS path
ip as-path access-list 1 permit ^10_60
!
! Define the access list
access-list 69 permit 10.69.0.0
```

Router B Configuration

```
router bgp 10
 neighbor 10.30.30.1 remote-as 30
 neighbor 10.30.30.1 send-community
 neighbor 10.30.30.1 route-map send_community out
!
ip bgp-community new-format
!
! Match prefix 10 and set community to 60:1
route-map send_community permit 10
 match ip address 10
 set community 60:1
!
! Match prefix 20 and set community to 60:2
route-map send_community permit 20
 match ip address 20
 set community 60:2
!
! Match prefix 30 and set community to 60:3
route-map send_community permit 30
 match ip address 30
 set community 60:3
!
! Match prefix 40 and set community to 60:4
route-map send_community permit 40
 match ip address 40
 set community 60:4
!
! Match prefix 50 and set community to 60:5
route-map send_community permit 50
 match ip address 50
 set community 60:5
!
! Match prefix 60 and set community to 60:6
route-map send_community permit 60
 match ip address 60
 set community 60:6
!
! Match prefix 70 and set community to 60:7
route-map send_community permit 70
 match ip address 70
 set community 60:7
!
! For all others, set community to 60:8
route-map send_community permit 80
 set community 60:8
```

```
!
! Define the access lists
access-list 10 permit 10.61.0.0
access-list 20 permit 10.62.0.0
access-list 30 permit 10.63.0.0
access-list 40 permit 10.64.0.0
access-list 50 permit 10.65.0.0
access-list 60 permit 10.66.0.0
access-list 70 permit 10.67.0.0
```

Examples Key Management

The following example configures a key chain named *trees*. In this example, the software will always accept and send *willow* as a valid key. The key *chestnut* will be accepted from 1:30 p.m. to 3:30 p.m. and be sent from 2:00 p.m. to 3:00 p.m. The overlap allows for migration of keys or discrepancy in the set time of the router. Likewise, the key *birch* immediately follows *chestnut*, and there is a 30-minute leeway on each side to handle time-of-day differences.

```
Router(config)# interface Ethernet 0
Router(config-if)# ip rip authentication key-chain trees
Router(config-if)# ip rip authentication mode md5
Router(config-if)# exit
Router(config)# router rip
Router(config-router)# network 172.19.0.0
Router(config-router)# version 2
Router(config-router)# exit
Router(config)# key chain trees
Router(config-keychain)# key 1
Router(config-keychain-key)# key-string willow
Router(config-keychain-key)# key 2
Router(config-keychain-key)# key-string chestnut
Router(config-keychain-key)# accept-lifetime 13:30:00 Jan 25 2005 duration 7200
Router(config-keychain-key)# send-lifetime 14:00:00 Jan 25 2005 duration 3600
Router(config-keychain-key)# key 3
Router(config-keychain-key)# key-string birch
Router(config-keychain-key)# accept-lifetime 14:30:00 Jan 25 2005 duration 7200
Router(config-keychain-key)# send-lifetime 15:00:00 Jan 25 2005 duration 3600
Router(config-keychain-key)# exit
```

The following example configures a key chain named trees:

```
Router(config)# key chain trees
Router(config-keychain)# key 1
Router(config-keychain-key)# key-string willow
Router(config-keychain-key)# key 2
Router(config-keychain-key)# key-string chestnut
Router(config-keychain-key)# accept-lifetime 00:00:00 Dec 5 2004 23:59:59 Dec 5 2005
Router(config-keychain-key)# send-lifetime 06:00:00 Dec 5 2004 18:00:00 Dec 5 2005
Router(config-keychain-key)# exit
Router(config-keychain)# exit
Router(config)# interface Ethernet 0
Router(config-if)# ip address 172.19.104.75 255.255.255.0 secondary 172.19.232.147
255.255.255.240
Router(config-if)# ip rip authentication key-chain trees
Router(config-if)# media-type 10BaseT
Router(config-if)# exit
Router(config)# interface Ethernet 1
Router(config-if)# no ip address
Router(config-if)# shutdown
Router(config-if)# media-type 10BaseT
Router(config-if)# exit
Router(config)# interface Fddi 0
Router(config-if)# ip address 10.1.1.1 255.255.255.0
Router(config-if)# no keepalive
Router(config-if)# exit
Router(config)# interface Fddi 1
Router(config-if)# ip address 172.16.1.1 255.255.255.0
Router(config-if)# ip rip send version 1
Router(config-if)# ip rip receive version 1
```

```
Router(config-if)# no keepalive
Router(config-if)# exit
Router(config)# router rip
Router(config-router)# version 2
Router(config-router)# network 172.19.0.0
Router(config-router)# network 10.0.0.0
Router(config-router)# network 172.16.0.0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
IP Routing Protocol-Independent commands	*Cisco IOS IP Routing: Protocol-Independent Command Reference*
IPv6 Routing: Static Routing	*IP Routing Protocol -Independent Configuration Guide*

Standards and RFCs

Standard/RFC	Title
No new or modified standards or RFCs are supported, and support for existing standards or RFCs has not been modified.	—

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring IP Routing Protocol-Independent Features

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 4 Feature Information for Configuring IP Routing Protocol-Independent Features

Feature Name	Releases	Feature Information
Default Passive Interface	12.0	In Internet service provider (ISP) and large enterprise networks, many of the distribution routers have more than 200 interfaces. Obtaining routing information from these interfaces required configuration of the routing protocol on all interfaces and manual configuration of the **passive-interface** command on the interfaces where adjacency was not desired. The Default Passive Interface feature simplifies the configuration of distribution routers by allowing all interfaces to be set as passive by default using a single **passive-interface default** command, and then by configuring individual interfaces where adjacencies are desired using the **no passive-interface** command.
Fast-Switched Policy Routing	11.3	IP policy routing can also be fast-switched. Prior to fast-switched policy routing, policy routing could only be process-switched, which meant that on most platforms, the switching rate was approximately 1000 to 10,000 packets per second. Such rates were not fast enough for many applications. Users that need policy routing to occur at faster speeds can now implement policy routing without slowing down the router.
IP Routing	11.0 Cisco IOS XE Release 3.1.0SG	The IP Routing feature introduced basic IP routing features that are documented throughout this document and also in other IP Routing Protocol documents.

Feature Name	Releases	Feature Information
NetFlow Policy Routing (NPR)	12.0(3)T	NetFlow policy routing (NPR) integrates policy routing, which enables traffic engineering and traffic classification, with NetFlow services, which provide billing, capacity planning, and monitoring information on real-time traffic flows. IP policy routing works with Cisco Express Forwarding (CEF), distributed CEF (dCEF), and NetFlow.
Policy-Based Routing	11.0	The Policy-Based Routing feature introduced a more flexible mechanism for routing packets than destination routing. Policy-based routing is a process where a router puts packets through a route map before routing the packets. The route map determines which packets are routed to which router next. The following command was introduced by this feature: **ip policy route-map**.
Policy-Based Routing (PBR) Default Next-Hop Route	12.1(11)E	The Policy-Based Routing (PBR) Default Next-Hop Route feature introduces the ability for packets that are forwarded as a result of the **set ip default next-hop** command to be switched at the hardware level. In prior releases, the router packets to be forwarded that are generated from the route map for PBR are switched at the software level. The following command was modified by this feature: **set ip default next-hop**.

Feature Name	Releases	Feature Information
Policy Routing Infrastructure	12.2(15)T	The Policy Routing Infrastructure feature provides full support of IP policy-based routing in conjunction with Cisco Express Forwarding (CEF) and NetFlow. As CEF gradually obsoletes fast switching, policy routing is integrated with CEF to increase customer performance requirements. When both policy routing and NetFlow are enabled, redundant processing is avoided.
QoS Policy Propagation via BGP	12.0	The QoS Policy Propagation via BGP feature allows you to classify packets by IP precedence based on BGP community lists, BGP autonomous system paths, and access lists. After a packet has been classified, you can use other QoS features such as committed access rate (CAR) and Weighted Random Early Detection (WRED) to specify and enforce policies to fit your business model.