CHAPTER 3

Managing Routing Protocols

Routing, introduced in Chapter 2, "Deploying Interior Routing Protocols," is crucial to the proper operation of your network. The routing function of a network is like the circulatory system of the human body: It is responsible for moving vital elements (data or blood cells) through a network efficiently. When those vital elements fail to reach their intended destinations, you can expect the systems that feed off the network to suffer (this applies to both internetworks and humans).

Chapter 2 covered basic routing configuration, but routing involves much more than simply enabling RIP, IGRP, OSPF, and EIGRP and letting them run. In a real-world network, routing protocols must be managed, extended, and optimized to promote overall network stability, flexibility, and efficiency. This chapter covers the key techniques that help you achieve these objectives.

The main topics of this chapter are

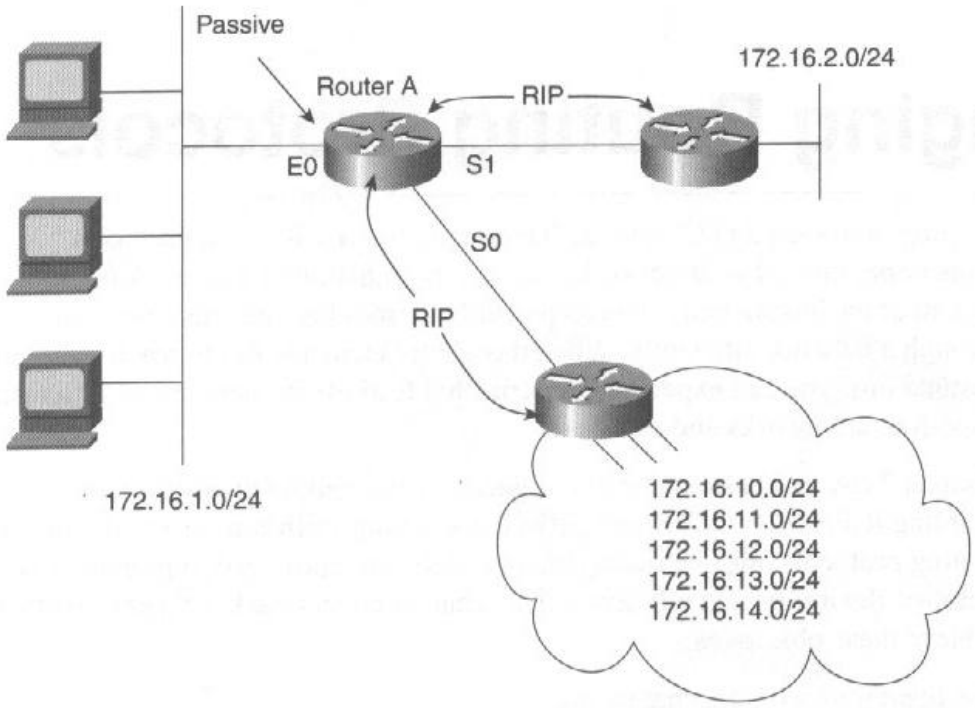
- Configuring Passive Interfaces
- Filtering Routing Updates
- Managing Redistribution
- Resolving Issues with VLSM and Classful Routing Protocols
- Leveraging Default Routing
- Configuring Route Summarization
- Deploying Policy Routing with Route Maps

Configuring Passive Interfaces

A common way of controlling routing information is to make an interface passive. A passive interface is a silenced interface: an interface on which you deliberately suppress the advertising of routing updates. You might want to do this in certain situations. For example, for security reasons you might have to block routing updates sent to a particular department or company because the updates reveal the topology of your network. In another case, when redistributing from one routing protocol to another, passive interfaces localize updates for efficiency and stability (see "Managing Redistribution," later in this chapter). Also, in dial-on-demand routing setups, passive interfaces prevent routing updates from triggering dial-up lines that are billed per minute—this controls operational costs.

A simple use of a passive interface is for silencing chatty protocols such as RIP on networks that do not require routing updates. Figure 3-1 illustrates such a scenario.

Figure 3-1 *Using a Passive Interface to Block Routing Information*



As depicted in Figure 3-1, Router A is using RIP to learn and advertise the subnets in major net 172.16.0.0. However, Router A is the only RIP device on the Ethernet LAN 172.16.1.0 (assume the clients are not RIP-enabled). Therefore, it is unnecessary and wasteful for Router A to send RIP updates out its Ethernet0 interface—no other routers are on the LAN and none of the clients care to receive any RIP information. By configuring Ethernet0 as passive, you can prevent all RIP advertisements from being sent out the interface and to the clients. The IOS command that does this is **passive-interface**, a router configuration mode command. The following is Router A's RIP configuration with the enhancement:

```
router rip
 network 172.16.0.0
 passive-interface Ethernet0
```

Router A is running RIP—it just isn't sending any RIP packets out Ethernet0. In fact, Router A still listens to RIP on the passive interface and would not block any RIP updates arriving on it. To prevent the router from listening to RIP, you must use route filters (see the following section, "Filtering Routing Updates").

To verify that the **passive-interface** command is working, you can look at the debugging messages for the routing protocol. Here's an output of **debug ip rip** for the scenario in Figure 3-1:

```
RTA#debug ip rip
RIP protocol debugging is on

RIP: sending v1 update to 255.255.255.255 via Serial0 (172.16.100.1)
     subnet  172.16.1.0, metric 1
     subnet  172.16.2.0, metric 2
     subnet  172.16.101.0, metric 1
RIP: sending v1 update to 255.255.255.255 via Serial1 (172.16.101.1)
     subnet  172.16.1.0, metric 1
     subnet  172.16.10.0, metric 3
     subnet  172.16.11.0, metric 3
     subnet  172.16.12.0, metric 4
     subnet  172.16.13.0, metric 4
     subnet  172.16.14.0, metric 4
     subnet  172.16.100.0, metric 1
```

The preceding output is one of Router A's periodic waves of RIP updates. The output confirms that Router A is sending updates from Serial0 and Serial1, but not from Ethernet0. Ethernet0 is noticeably absent from the debug output.

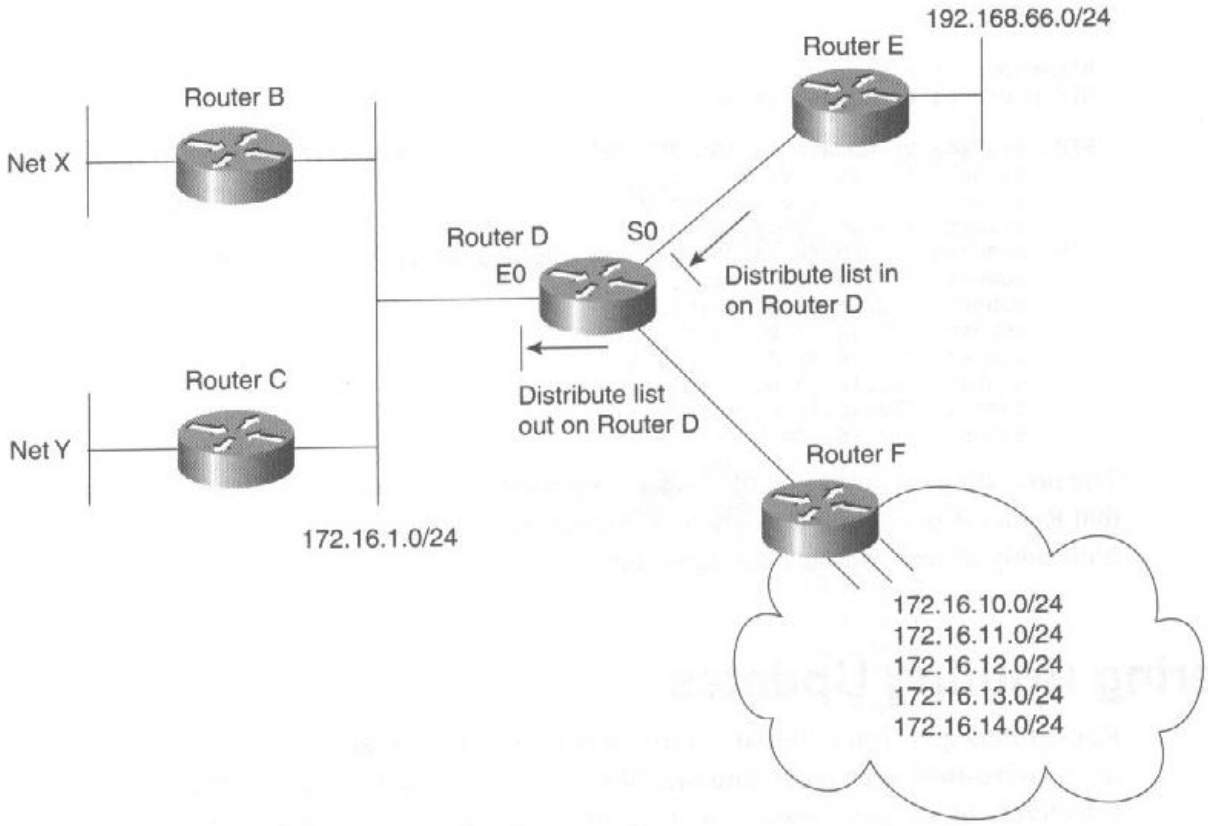
Filtering Routing Updates

Route filters give you granular control over the routes sent and received by your router. Unlike the **passive-interface** command that blocks all routes sent out an interface, a route filter can selectively block some updates and let others through. Route filters can also block incoming routes—something the **passive-interface** command cannot do. Filtering incoming routes is like filtering e-mail spam. It rejects routes that are unwanted or unnecessary (for example, improperly sourced default routes that might confuse your router).

Like the passive interface, route filtering is a building block for routing protocol redistribution (which is why it is covered in this chapter before the section on redistribution). The technique is also useful for filtering and sanitizing routes received from a router outside of your control—a router managed by another department or organization, for example.

Suppose you need to control the dispersion of routing updates in the network depicted in Figure 3-2 (the routing protocol is IGRP).

Figure 3-2 *A Scenario for Configuring Route Filters*



The situation you face is the following:

- You manage all routers except Router E. Router E is owned by another group who often reconfigure, or rather misconfigure, their router. This sometimes causes problems for your router, Router D, because it gets confused when it receives illegal routes from Router E. You know that one route, 192.168.66.0/24, should be accepted by Router D. All other routes from Router E are noise and should be filtered.
- For security reasons, Router B and Router C should not have a route to subnet 172.16.13.0/24. This prevents users in Net X and Net Y from accessing that particular subnet. Subsequently, Router D needs to filter 172.16.13.0/24 from its updates to Router B and Router C.

To accomplish these objectives, configure route filters on Router D with the **distribute-list** router mode command. The following is Router D's configuration:

```
router igrp 100
 network 172.16.0.0
 distribute-list 1 in Serial0
 distribute-list 2 out Ethernet0
!
access-list 1 permit 192.168.66.0 0.0.0.255
access-list 2 deny   172.16.13.0 0.0.0.255
access-list 2 permit any
```

The command **distribute-list 1 in Serial0** tells Router D to filter all IGRP updates inbound on Serial0 based on the criteria defined by access list 1, which permits route 192.168.66.0/24 only. This is the only route Router D will accept from Router E. All other routes entering Serial0 (sent from Router E) are ignored.

The command **distribute-list 2 out Ethernet0** tells the router to filter all IGRP updates outbound on Ethernet0, based on the criteria defined by access list 2. Access list 2 denies route 172.16.13.0/24 and permits all other routes—thus meeting the stated requirement to prevent Router B and Router C from learning route 172.16.13.0/24.

The remaining **access-list** commands define the access lists, 1 and 2, that are referenced by the **distribute-list** commands. You don't have to configure a **deny** rule for access list 1 because of the invisible deny-any rule at the end of every access list. Access lists and their syntax are covered in Chapter 6, "Deploying Basic Security Services."

After configuring the route filters, you should verify that they are delivering the expected results. You can issue **show ip route** on Router D to ensure that Router D is accepting only route 192.168.66.0/24 from Router E. Likewise, you can issue **show ip route** on Router B and Router C and verify that those routers have all routes except 172.16.13.0/24.

You can also verify route filtering with debugging commands. The following is an output with **debug ip igrp transactions** enabled on Router D:

```
RTD#deb ip igrp tr
IGRP protocol debugging is on

IGRP: sending update to 255.255.255.255 via Ethernet0 (172.16.1.1)
      subnet 172.16.2.0, metric=501
      subnet 172.16.3.0, metric=501
      subnet 172.16.10.0, metric=9675
      subnet 172.16.11.0, metric=9675
      subnet 172.16.12.0, metric=10255
      subnet 172.16.14.0, metric=9675
      network 192.168.66.0, metric=8576
```

The preceding output shows the IGRP updates sent by Router D out its Ethernet0 interface. Noticeably absent from the list is route 172.16.13.0/24, the route blocked by the route filter (**distribute-list 2 out Ethernet0**). Subnets 172.16.2.0 and 172.16.3.0 are the links joining Router D to Routers E and F (refer to Figure 3-2).

NOTE OSPF complicates the use of route filters somewhat and, in general, you should avoid OSPF route filtering whenever possible. With OSPF, you cannot use **distribute-list out** and specify an interface; however, when redistributing with OSPF, you can specify an external routing protocol with **distribute-list out**. You can prevent a router from entering a route in its routing table with **distribute-list in**, but this does not stop the router from forwarding the LSA to other routers in the network. Because OSPF is a link-state protocol and requires extensive propagation of LSAs, it cannot be route filtered like RIP, IGRP, and EIGRP.

Managing Redistribution

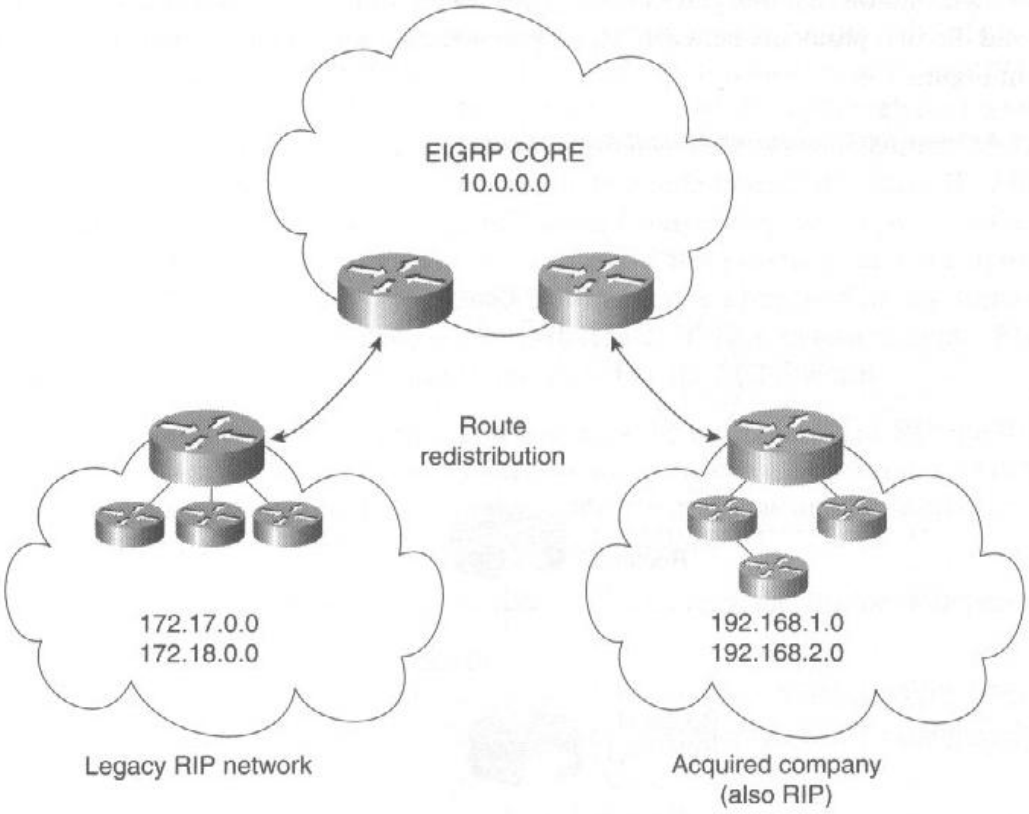
Ideally, all routers in your organization should run the same routing protocol. In reality, however, doing so can be difficult, especially when you are faced with any of the following situations:

- Various groups are managing their own routers and must use different routing protocols for one reason or another. Yet everyone must be internetworked together.
- You are migrating a large network of routers to a new routing protocol but you don't want to convert all routers at once. Instead, you want to migrate sections of the network in phases. During the migration, routers using the new protocol must still interoperate with routers using the old protocol.
- You have just acquired or merged with another organization and it doesn't use the same routing protocol. You need to establish connectivity between the organizations now. Later on, you might consider migrating the unified network to a single routing protocol.
- You run multiple autonomous systems or processes of the same routing protocol for administrative purposes and you need to share routing information across these routing domain boundaries.
- You need to exchange routing information with your Internet Service Provider, who runs a different routing protocol than you do.

Routing protocol redistribution resolves these issues. It is a translation service that allows different routing protocols to interoperate. Consider, for example, an organization's network depicted in Figure 3-3.

As shown in Figure 3-3, this organization has deployed EIGRP (OSPF works equally well) on a core group of routers to take advantage of such varied features as VLSM, fast convergence, better scalability, and route summarization. Off to the lower left is a legacy RIP network—these might be routers that can run only RIP or they might be routers that haven't yet migrated from RIP to EIGRP. To the lower right is a newly acquired company that needs connectivity to the rest of the organization. It also uses RIP.

Figure 3-3 *A Scenario for Routing Protocol Redistribution*



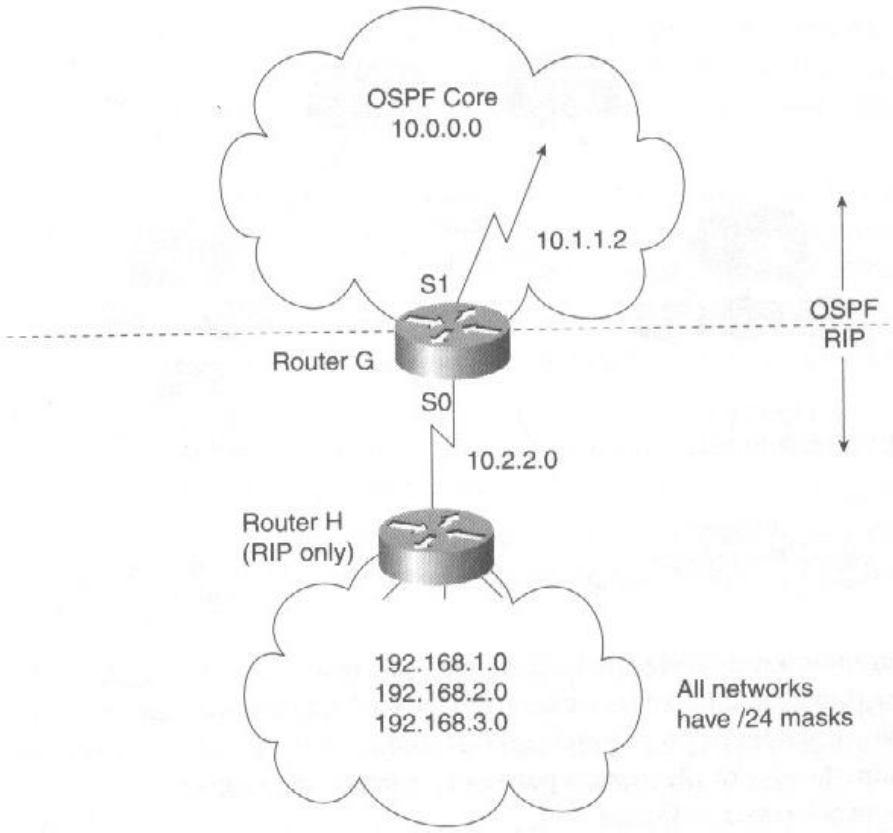
Redistribution is a service that allows the sharing of routes across the various RIP and EIGRP routing domains. It transfers routes from the RIP domain to the EIGRP domain, and vice versa, so that every route is propagated and full connectivity is achieved (assuming there are no route filters in place). For illustration purposes, consider the propagation of the routes in the legacy RIP network (refer to Figure 3-3):

1. Routes from the legacy RIP network (172.17.0.0 and 172.18.0.0) are redistributed into EIGRP. The routes are converted from the RIP format to an EIGRP format and injected into the core.
2. The routes propagate through the core as EIGRP routes until they reach the boundary between the core and the acquisition network (right side of Figure 3-3).
3. Routes 172.17.0.0 and 172.18.0.0 are redistributed again, but this time from EIGRP into RIP. As the routes are injected into the acquisition network, they are converted from EIGRP to RIP.
4. Routes 172.17.0.0 and 172.18.0.0 are accepted and propagated through the acquisition network as RIP routes.
5. The result: Routers in the core and in the acquisition network now know how to reach 172.17.0.0 and 172.18.0.0 because of redistribution. A similar exercise can be done for tracing the redistribution of routes sourced by the acquisition network.

Configuring Redistribution—RIP and OSPF

Redistribution requires you to configure a router with the **redistribute** router mode command and the two protocols between which you want to redistribute. Consider the example network in Figure 3-4.

Figure 3-4 *A Network for Configuring Mutual Redistribution*



In the example, Router H is a RIP-only router with RIP routes 192.168.x.0 (where x ranges from 1 to 3). Router G is the redistribution router: It must run both RIP and OSPF.

The routing configuration for Router H is nothing new:

```
router rip
 network 10.0.0.0
 network 192.168.1.0
 network 192.168.2.0
 network 192.168.3.0
```

Router G, on the other hand, is a redistribution router. The following is Router G's RIP configuration:

```
router rip
 redistribute ospf 10 metric 3
 passive-interface Serial1
 network 10.0.0.0
```

The command **redistribute ospf 10 metric 3** redistributes the routes Router G learned from its OSPF process 10 into its RIP process with a default metric of 3 hops. Because OSPF has a different metric than RIP, you must assign a starting metric for the redistributed routes (keywords **metric 3**). The **redistribute** command injects a lot of routes into RIP—you can imagine there are many routes up there in that OSPF core. It's as though the router has two "brains": one that knows everything about the OSPF world and another that knows everything about the RIP world. When you redistribute from OSPF into RIP with **redistribute ospf 10 metric 3** (a subcommand of **router rip**), you copy all of the contents of the OSPF brain into the RIP brain. Subsequently, this provides RIP with all of the information learned from OSPF, and that information is a bunch of routes. Router G then advertises the combined database of routes to Router H. This, in turn, provides Router H (a RIP-only router) with all of the routes originated by the OSPF world.

The command **passive-interface Serial1** disables the transmission of RIP out Router G's Serial1 interface. Because only OSPF routers are in the core, there is no need to send RIP advertisements out this interface—disable it for less overhead and more efficiency (see "Configuring Passive Interfaces," earlier in this chapter).

The command **network 10.0.0.0** is the basic RIP command that enables RIP processing on major net 10.0.0.0.

Router G also needs to redistribute routes the other way: from RIP into OSPF. This is called two-way or *mutual redistribution*. The following is Router G's OSPF configuration:

```
router ospf 10
 redistribute rip metric 50 metric-type 1 subnets
 network 10.1.1.0 0.0.0.255 area 0
```

The command **redistribute rip metric 50 metric-type 1 subnets** redistributes the routes learned by the RIP process into OSPF process 10. Break up the command by keywords and you get the following:

- **rip**: The source for the redistribution is the RIP process. Notice that this time **redistribute** is a subcommand of **router ospf 10**. When you want to redistribute into a routing protocol, you put the **redistribute** command under the **router** command for that protocol.
- **metric 50**: Because RIP uses hop count and OSPF uses cost, you have to specify a starting (or default) metric for the routes redistributed into OSPF. Here, the default metric is a cost of 50.
- **metric-type 1**: This sets the redistributed routes to OSPF external type 1 routes. This applies only when you redistribute into OSPF. Basically, two types of external (redistributed) routes exist in OSPF: type 1 and type 2. The metric for type 1 routes is the sum of the cost to reach the redistribution router and the default metric. The metric for type 2 routes is simply the default metric—the OSPF cost is not added because the route propagates the OSPF domain. If you want to include the OSPF cost to reach the redistribution point, use external type 1 routes.

- **subnets**: This tells the router to redistribute both major nets and their subnets. If you exclude this, only major nets are redistributed. This applies only when redistributing into OSPF. A common error is to forget the **subnets** keyword when redistributing into OSPF.

Router G's OSPF configuration does not need the **passive-interface** command because only Serial1 is assigned to an OSPF area. Serial0 is not in an area, and therefore, does not send any OSPF updates. Consult Chapter 2 for OSPF configuration.

A look at the routing table of a core OSPF router (a router somewhere in the OSPF cloud in Figure 3-4) proves that the RIP routes are being redistributed into OSPF:

```
CoreRouter#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
       T - traffic engineered route

Gateway of last resort is not set

     192.168.11.0/24 is subnetted, 1 subnets
O IA    192.168.11.1 [110/51] via 10.3.3.2, 00:30:40, Serial1
     10.0.0.0/24 is subnetted, 2 subnets
O E1    10.2.2.0 [110/320] via 10.3.3.2, 00:30:40, Serial1
C       10.3.3.0 is directly connected, Serial1
O E1 192.168.1.0/24 [110/400] via 10.3.3.2, 00:30:40, Serial1
O E1 192.168.2.0/24 [110/400] via 10.3.3.2, 00:29:54, Serial1
O E1 192.168.3.0/24 [110/460] via 10.3.3.2, 00:29:54, Serial1
<lines deleted for brevity>
```

In the preceding output, the lines highlighted in boldface are the routes originally sourced from the RIP domain and redistributed into OSPF by Router G. The code **O E1** designates an OSPF type 1 external route. An external route is simply a route that was redistributed into OSPF.

Redistributing into IGRP and EIGRP

Redistributing into IGRP and EIGRP is similar to redistributing into RIP and OSPF except you must define each variable of the composite metric to arrive at the default metric:

```
router eigrp 100
 redistribute rip metric 1544 100 255 1 1500
 network 172.16.0.0
```

where the numbers in the **redistribute** command after the keyword **metric** specify:

- **Bandwidth**—In kilobits per second.
- **Delay**—In tens of microseconds. That is, **100** equals a hundred "tens" or 1,000 microseconds.

- **Reliability**—A number from 0 to 255, where 255 is most reliable.
- **Loading**—A number from 0 to 255, where 255 is a fully loaded (100% saturated) link.
- **MTU (Maximum Transmission Unit)**—In bytes.

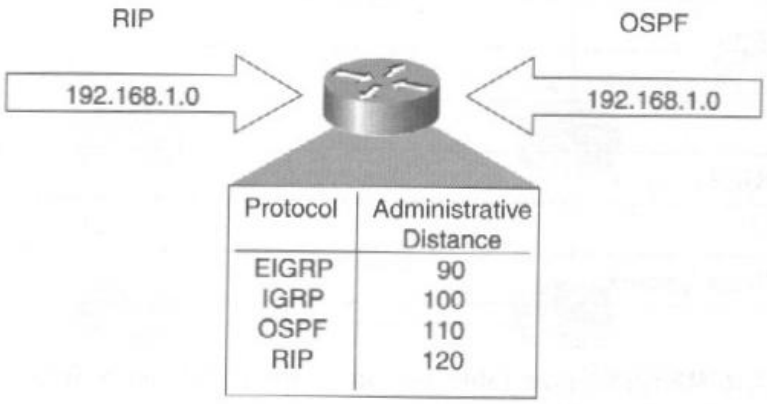
For more information on IGRP and EIGRP metrics, see the sources listed in the Bibliography and at http://www.cisco.com/warp/public/103/index.shtml (or search for "igrp metric" on the Cisco Web site).

NOTE IOS automatically redistributes between IGRP and EIGRP when the autonomous system numbers are the same. You don't have to enter the **redistribute** command. It also adjusts the metrics as it redistributes, because the EIGRP metric is simply 256 multiplied by the IGRP metric (EIGRP metric = IGRP metric × 256).

Understanding Administrative Distance

When you use more than one routing protocol—for example, when you redistribute—you need to be aware of *administrative distances*. An administrative distance is a priority assigned to a route based on its routing protocol. When a router receives the same route from more than one protocol, which one should it use? Should it, for example, use the route heard from RIP or the one heard from OSPF? Because RIP and OSPF have completely different metrics, the router cannot decide by comparing metrics, so another value, the administrative distance, is used. Consider the diagram in Figure 3-5.

Figure 3-5 *Selecting Routes with Administrative Distance*



The router in Figure 3-5 has learned the same network, 192.168.1.0, on two different interfaces and from two different protocols: RIP and OSPF. To determine the route that should be used, the router looks at its preprogrammed ranking of protocols by administrative distance. The route learned by the protocol with the lowest administrative distance is preferred. It is the route installed in the routing table and used to reach the destination. For the situation depicted in Figure 3-5, the router prefers the OSPF route to the RIP route because OSPF has a lower administrative distance (110 for OSPF versus 120 for RIP).

NOTE Administrative distances are locally significant: They are not communicated across routers.

The default administrative distance for each routing protocol is listed in Table 3-1. By default, all routes learned from a protocol have the same administrative distance—for example, all OSPF routes have an administrative distance of 110.

Table 3-1 *Default Administrative Distances by Routing Protocol*

Type of Route	Administrative Distance
Connected interface	0
Static route	1
EIGRP summary route	5
External BGP	20
EIGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
External EIGRP	170
Internal BGP	200
Unknown/Route ignored	255

External EIGRP routes (see Table 3-1) are routes redistributed into EIGRP. For example, RIP routes redistributed into EIGRP are marked as external routes and carry this distinction as they are routed by EIGRP.

To change the administrative distance of a routing protocol, issue the **distance** router mode command:

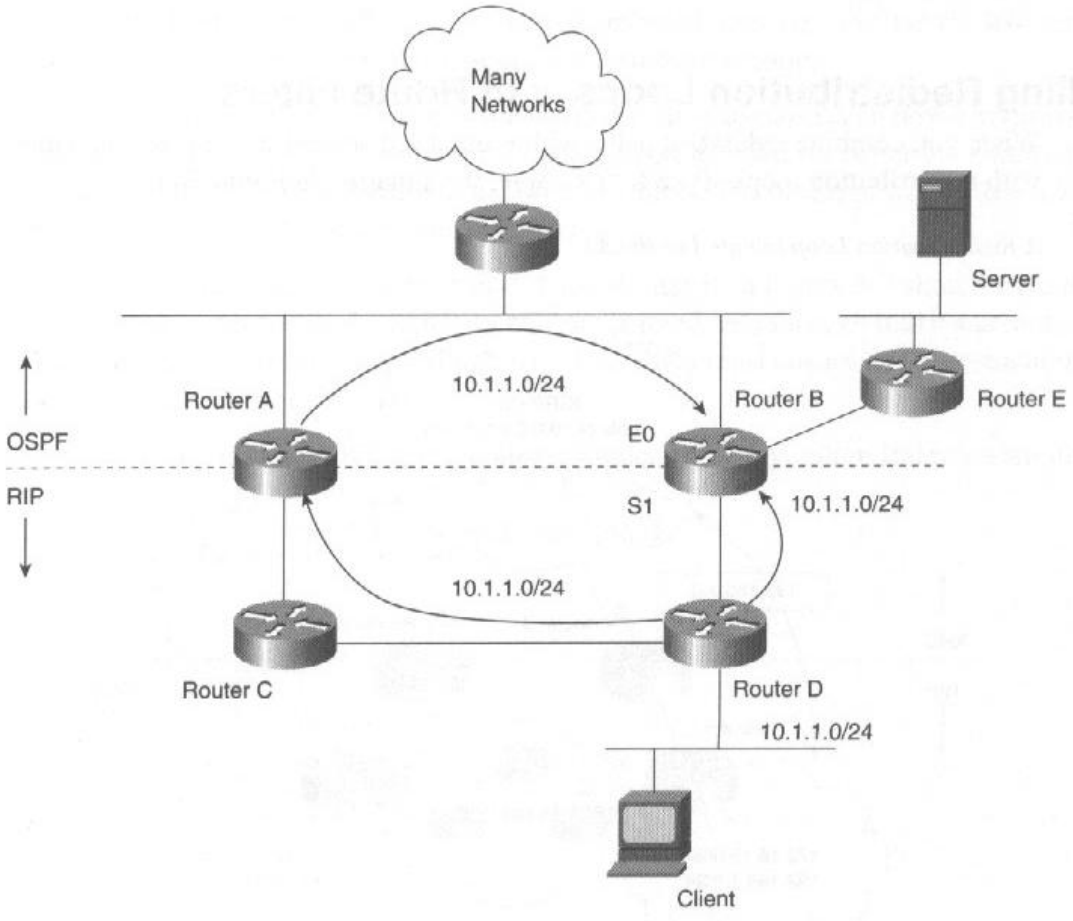
```
router ospf 10
 distance 125
```

The preceding configuration changes the administrative distance of OSPF routes from 110 to 125. This is higher than RIP's administrative distance and means the router will prefer RIP routes to OSPF routes (RIP has a default distance of 120). Altering the default distances like this is often done during a migration to a new routing protocol—that is, you make the new routing protocol a higher distance than the old protocol until the new protocol is stable. After the new protocol is in place, you can return the distances to their original values and then disable the old protocol.

NOTE The administrative distance of a route is sometimes called a measure of the route's *believability*. A router considers a route more believable than another if it has a lower administrative distance.

Administrative distances can cause problems when you redistribute. Consider the network in Figure 3-6.

Figure 3-6 *Problems with Administrative Distance and Redistribution*



In Figure 3-6, Router A and Router B are doing mutual redistribution between the OSPF and RIP domains. Router D sends RIP advertisements for its directly connected network 10.1.1.0/24. Router C forwards the route to Router A, which redistributes it into OSPF. The route then travels over OSPF to Router B (Router B accepts the route through its Ethernet0 interface). Router B has just learned the route 10.1.1.0/24 through OSPF. Now, Router B also learns of 10.1.1.0/24 on its Serial1 interface through RIP—the route is sent by Router D, as depicted by the arrow from Router D to Router B in Figure 3-6.

Of the two routes received, Router B uses the route learned through OSPF because it has a lower administrative distance than RIP. Subsequently, all traffic from Router B to 10.1.1.0/24 follows a roundabout path (B→A→C→D) instead of the more direct path from Router B to Router D. When the server near Router E (refer to Figure 3-6) sends data to the client below Router D, the traffic flows along the suboptimal path because Router B selects Router A as the next hop. This is an undesirable consequence of redistribution. One way to solve the problem is to apply an inbound route filter on Router B such that Router B does not install the OSPF route for 10.1.1.0/24 in its routing table, thus forcing Router B to use the RIP route to reach the destination. However, this solution does not stop Router B from forwarding the route to other OSPF routers, because OSPF is a link-state protocol and must broadly propagate LSAs. The following section looks at route filters and redistribution in more detail.

Controlling Redistribution Loops with Route Filters

When you combine redundant paths with mutual redistribution, you can encounter problems with redistribution loops. Take for example, the situation in Figure 3-7.

Figure 3-7 *A Redistribution Loop (Route Feedback)*

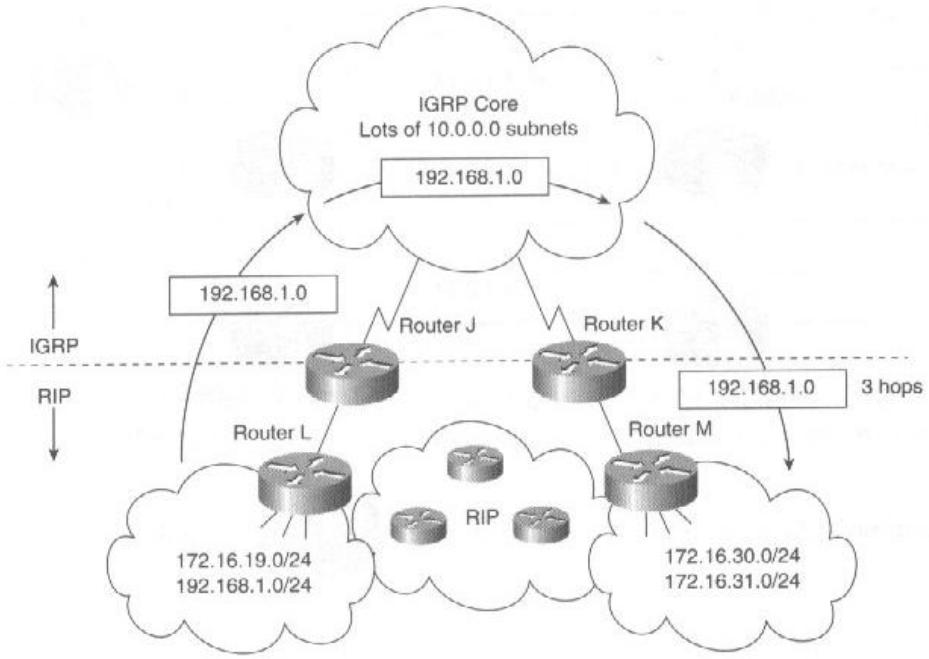


Figure 3-7 shows two points along the IGRP-RIP boundary where redistribution is occurring. Both Router J and Router K are performing mutual redistribution between RIP and IGRP.

Care must be taken to ensure that a route redistributed from RIP into IGRP at Router J does not get redistributed into RIP at Router K—an effect called *route feedback*. Route feedback is represented by the propagation of route 192.168.1.0/24 in Figure 3-7. Here's the sequence:

1. Route 192.168.1.0/24 originates from the RIP cloud behind Router L, propagates up to Router J, and is redistributed into IGRP by Router J.
2. In the IGRP domain, the route propagates via IGRP to Router K, where it is redistributed into RIP (Router K sees 192.168.1.0/24 as just another IGRP update) with a low default metric. Suppose the metric is 3 hops.
3. In turn, Router K forwards the route to Router M with a hop count of 3 (a low default metric), effectively telling Router M that the way to 192.168.1.0/24 is through Router K.
4. The undesirable result: Router M thinks the path to 192.168.1.0/24 is through Router K—a suboptimal route caused by the feedback of the route at Router K. Instead, Router K should reach the destination via the RIP cloud between it and Router L.

Using the same logic, you can deduce that IGRP routes are equally susceptible to route feedback. That is, an IGRP route can be redistributed into RIP, traverse the RIP domain, and get redistributed into IGRP at another redistribution point.

In addition to suboptimal routing, route feedback can also cause such problems as routing loops and loss of reachability. Therefore, it's highly recommended (or better yet *required*) that you always control mutual redistribution with route filters. The basics of route filters are covered in the earlier section "Filtering Routing Updates."

To prevent route feedback, both Router J and Router K in Figure 3-7 should have filters to ensure that routes within the IGRP domain are learned only through IGRP and routes within the RIP domain are learned only through RIP. The filters must ensure a RIP destination is not learned via IGRP and vice versa.

The following is Router K's configuration (Router J's configuration follows a similar strategy):

```
router igrp 100
 redistribute rip metric 1544 2000 255 1 1500
 network 10.0.0.0
 distribute-list 1 in
!
router rip
 redistribute igrp 100 metric 3
 network 172.16.0.0
 distribute-list 2 in
!
access-list 1 permit 10.0.0.0 0.255.255.255
access-list 2 permit 192.168.1.0 0.0.0.255
access-list 2 permit 172.16.0.0 0.0.255.255
```

In the preceding configuration, Router K is redistributing RIP into IGRP and IGRP into RIP (mutual redistribution). This is defined by the two **redistribute** commands.

The command **distribute-list 1 in** is a subcommand of the IGRP routing process activated by **router igrp 100**. It tells the router to filter all routes it receives through IGRP with access list 1. Access list 1 permits routes 10.*x*.*x*.*x* (where *x* denotes any octet) and blocks all other routes. This stops the feedback loop: Router K accepts only 10.0.0.0 routes (native IGRP routes) via IGRP and rejects any non-10.0.0.0 routes it learns via IGRP—routes that have been redistributed from RIP into IGRP.

NOTE In a real-world network, you typically have many more routes to permit in the access list than a single, contiguous 10.0.0.0. Ensure that your access list includes all routes on one side of the redistribution boundary. Otherwise, those routes will not be redistributed and users on the other side of the boundary will not be able to reach those destinations.

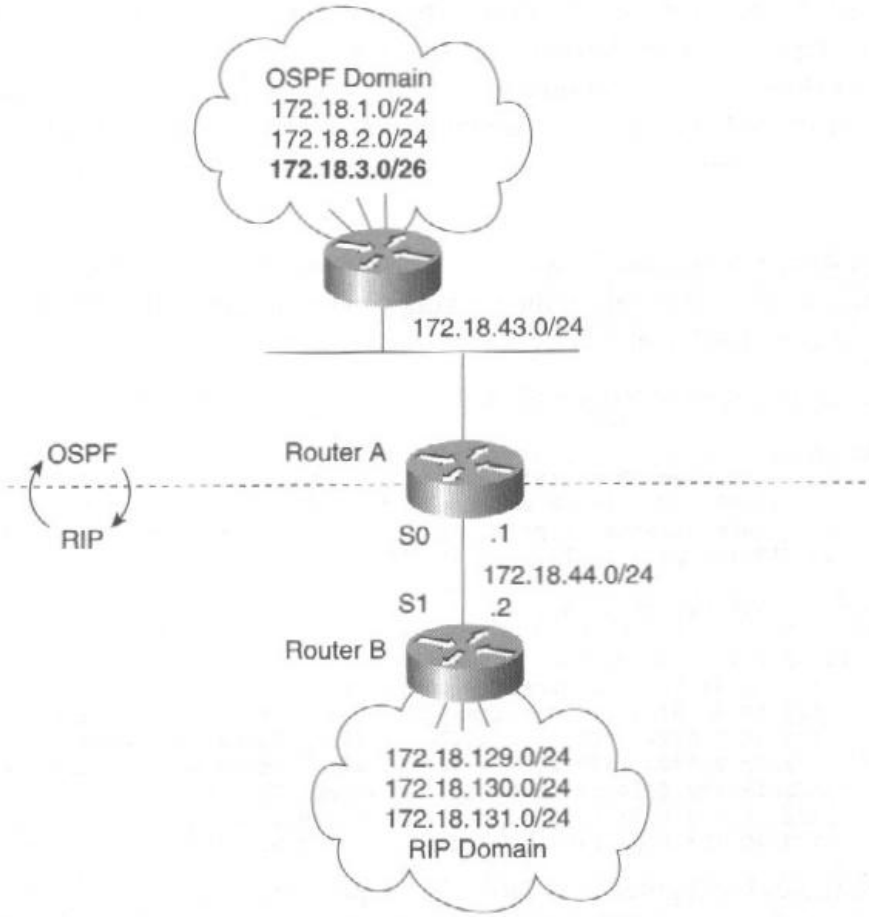
The command **distribute-list 2 in** (a subcommand of **router rip**) tells the router to filter all routes it receives through RIP with access list 2. Access list 2 permits routes 192.168.1.*x* and 172.16.*x*.*x* (where *x* denotes any octet) and blocks all other routes. Like the **distribute-list** command under IGRP, this prevents route feedback: Router K filters and accepts only native RIP routes via the RIP protocol.

The remaining commands, starting with **access-list**, define access lists 1 and 2. Access lists are covered in Chapter 6.

Resolving Issues with VLSM and Classful Routing Protocols

As mentioned in Chapter 1, "Managing Your IP Address Space," classful routing protocols such as RIP and IGRP do not support VLSM, but sometimes you are forced to resolve VLSM and classful routing incompatibilities in a real-world network. Especially when you redistribute, routes might get lost between the parts of your network that use VLSM and the other parts that are classful. Figure 3-8 illustrates a typical scenario.

Figure 3-8 *Issues with VLSM and Redistribution: An Example Network*



In Figure 3-8, Router A is performing mutual redistribution between OSPF and RIP, and Router B is a RIP-only router. Because RIP is classful and does not support VLSM, all networks in the RIP domain are properly deployed with the same mask, /24. The OSPF domain contains subnets from the same major net and uses VLSM. Subnet 172.18.3.0 has a /26 mask (highlighted in boldface) and all the other subnets in the OSPF domain have a /24 mask.

When Router A redistributes the routes from the OSPF domain into RIP, RIP advertises all of the routes out Serial0 except 172.18.3.0/26. It doesn't forward 172.18.3.0/26 because the mask for that route is not what Router A's RIP process expects. Router A examines the major net and mask configured on Serial0 *before* it advertises a route through it. If the route is part of the *same* major net as the interface but has a different mask, RIP does not advertise it out the interface (this rule applies to IGRP also).

NOTE If the route is from a *different* major net than the interface, the major net of the route is advertised out the interface (172.18.0.0/16, for example). In a classful world, the existence of a different major net means the router sits on a boundary between major nets. One major net does not need to know the internal subnets of another because subnets are contained within a major net. As mentioned in Chapter 2, classful protocols do not support discontiguous subnets.

Because Router A does not forward the VLSM route 172.18.3.0/26, Router B never receives it and no one in the RIP half of the network can reach subnet 172.18.3.0/26. The route is lost because it is redistributed into a classful routing protocol.

The following output of Router B's routing table shows the problem:

```
RTB#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default

Gateway of last resort is not set

     172.18.0.0 is subnetted, 7 subnets
C       172.18.44.0/24 is directly connected, Serial1
R       172.18.43.0/24 [120/1] via 172.18.44.1, 00:00:08, Serial1
R       172.18.1.0/24 [120/2] via 172.18.44.1, 00:00:08, Serial1
R       172.18.2.0/24 [120/2] via 172.18.44.1, 00:00:08, Serial1
C       172.18.129.0/24 is directly connected, Ethernet0
R       172.18.130.0/24 [120/4] via 172.18.129.2, 00:00:23, Ethernet0
R       172.18.131.0/24 [120/5] via 172.18.129.2, 00:00:23, Ethernet0
```

The two routes highlighted in boldface in the preceding output are the routes that Router A redistributes from the OSPF domain into RIP. Notice that route 172.18.3.0/26 is absent from the table because Router A is not forwarding it to Router B.

To fix the problem of the missing route, you can configure a static route on Router B that tells the router how to reach the lost subnet:

```
RTB#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
RTB(config)#ip route 172.18.3.0 255.255.255.192 172.18.44.1
```

The command **ip route 172.18.3.0 255.255.255.192 172.18.44.1** installs a static route in Router B. It tells the router to reach 172.18.3.0/26 through the next hop router, 172.18.44.1 (Router A). This fixes the problem for Router B, but it does not help other RIP routers in the RIP domain. You would have to install static routes on the other RIP routers—a laborious task when there are many routers or many destinations, so consider the next approach.

An alternate solution is to create a static route on Router A that is agreeable to RIP and then redistribute it into the RIP domain. The following is Router A's configuration with this approach (boldface highlights the new commands):

```
router rip
 redistribute static
 redistribute ospf 10 metric 3
 passive-interface Ethernet0
 network 172.18.0.0
!
ip route 172.18.3.0 255.255.255.0 Null0
```

where **redistribute static** tells the router to redistribute all static routes configured on the router into RIP. This is like redistributing OSPF into RIP except the source of the redistribution is static routing instead of OSPF.

The command **ip route 172.18.3.0 255.255.255.0 Null0** configures a static route in Router A. This and other existing static routes are redistributed into RIP by **redistribute static**. Notice a couple of things about the route:

- The route is cleverly crafted with a /24 mask—the same mask as Router A's Serial0 interface and the rest of the RIP domain. This route matches the mask that is consistent throughout the RIP domain, so it can be redistributed and circulated by RIP without any problems.
- The route points to the router's Null0 interface—a bit bucket that leads to nowhere. The section on "Verifying EIGRP Configuration" in Chapter 2 explains the strategy behind the Null0 interface. Briefly, Router A doesn't use this route to reach 172.18.3.0/26 because it uses the most specific route: the one it learned through OSPF (172.18.3.0/26 is more specific than 172.18.3.0/24). You create this static so that Router A can redistribute it and make it known to other routers. This is called *originating a static route*.

When Router A redistributes the static route into RIP, RIP sends it out Serial0 (it matches Serial0's /24 mask). Router B learns that it can reach 172.18.3.0/24 through Router A and then propagates the route as a normal RIP route to the rest of the RIP domain. Router B and other RIP routers can now reach 172.18.3.0/26 via the route you fabricated, 172.18.3.0/24.

A caveat of this approach is the generalization (or summarization) of 172.18.3.0/26 by 172.18.3.0/24. The actual subnet (/26 mask) is a subset of the larger address block defined by the static with the /24 mask. This is fine if all addresses in the larger block are in the OSPF domain and reachable from the RIP domain via Router A. However, if portions of 172.18.3.0/24 other than 172.18.3.0/26 (for example, 172.18.3.128/26) are not advertised into RIP by Router A, this approach could cause reachability problems.

NOTE You can see that there's quite a bit of subnetting and masking in this section. Refer to Chapter 1 if necessary.

Instead of originating a static route, an alternate solution is to use OSPF's summarization capabilities. With this approach, you use the OSPF **area range** command to summarize 172.18.3.0/26 with 172.18.3.0/24 (see "Configuring OSPF Summarization Between Areas" later in this chapter for use of this command). This gives you the same route 172.18.3.0/24, which you can then redistribute into RIP without a fuss.

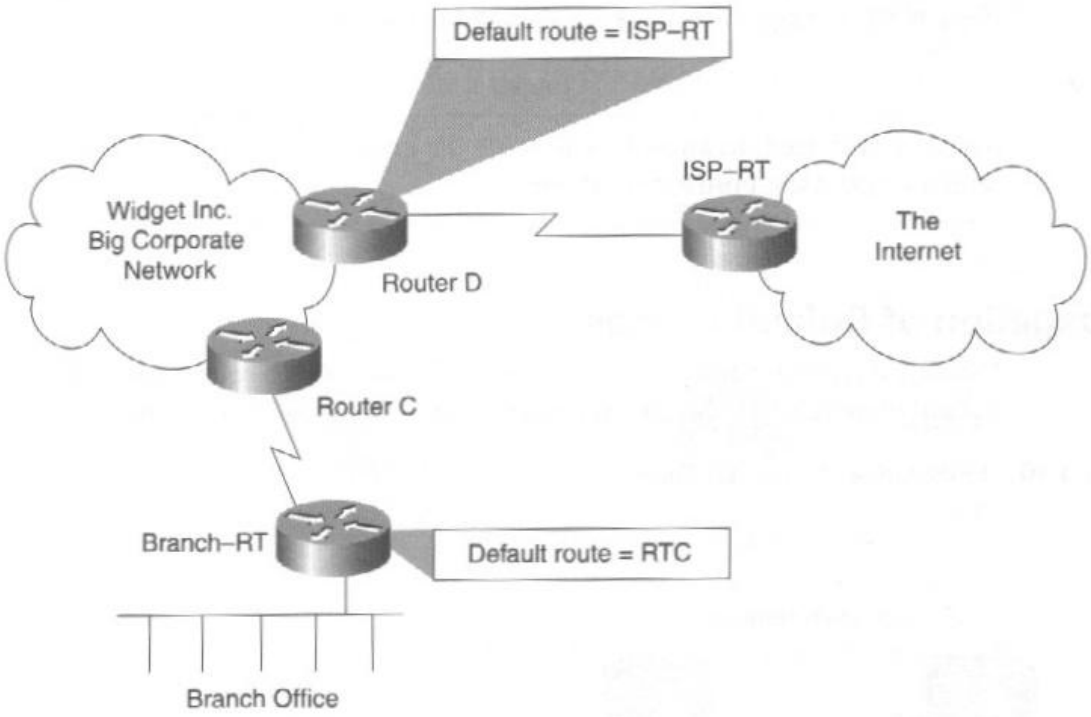
Sometimes, there is no perfect solution for resolving VLSM issues with classful routing protocols. If your network does not lend itself to a clean solution because the classless space is heavy with VLSM, you might have to use a combination of the approaches presented in this section. You might, for example, have to originate a static route and then fill in missing information by configuring static routes on some selected RIP routers. Alternatively, you might find a reasonable opportunity to renumber some subnets so you can originate a static route as in the example. Finally, you always have the option to migrate off the classful protocol or at least reduce its coverage.

Leveraging Default Routing

Default routing scales the network, conserves resources, and simplifies routing information. A *default route* is a special route that tells the router how to reach unknown destinations—that is, destinations that are absent from the routing table because they are neither learned through a routing protocol nor manually configured with static routes. A default route is a catchall: When the router doesn't know how to forward a packet (because the packet is destined for an unknown network), the router sends it to the next hop defined by the default route. Without the default route, the packet is dropped because the router has no idea how to forward it.

It might sound as though default routes exist to cover up routing mistakes, but that is not the case. With default routing, you purposely withhold routing information from the router so that its routing task is reduced and simplified. This reduces the size of the routing table and conserves router resources (memory and CPU). To make up for the missing routes (the missing information), you provide the router with a simple default route. It's like taking a burden off the router and simplifying its view of the world to only what it needs to know. Default routes can substantially reduce the overhead of routing protocols, and they are a necessity when you connect to the public Internet. Figure 3-9 describes the typical uses for default routing.

Figure 3-9 *Default Routing and the Network of Widget, Inc.*



In Figure 3-9, Router D at Widget, Inc., is connected to the Internet through an ISP router, ISP-RT. Because Widget has only one connection to the Internet, Router D doesn't need to hold and manage a full Internet routing table—that's a lot of routes. Router D has only one way to get to the Internet: through ISP-RT. Router D already knows how to reach everything inside of Widget through the interior routing protocol, so traffic going anywhere else must be destined for the Internet. Widget configures a default route on Router D that points to ISP-RT (the IOS configuration of default routes follows in later sections). When Router D receives a packet destined for a non-Widget network, it uses the default route and sends it to ISP-RT.

NOTE Even if Router D maintains a full Internet table because it's directly connected to the ISP, it's not practical for *all* routers in the big corporate network to also maintain all Internet routes. Eventually, routers in the corporate network downstream of Router D will require default routing to reach the Internet.

Figure 3-9 includes another default routing situation. The branch office router, Branch-RT, has a locally attached LAN and one path out to the big corporate network. Branch-RT doesn't need a full-size routing table with thousands of routes from the big corporate network because two choices suffice: Either packets go to the branch office LAN or they go somewhere else in the world via Router C. Therefore, Widget, Inc., configures a default route on Branch-RT that

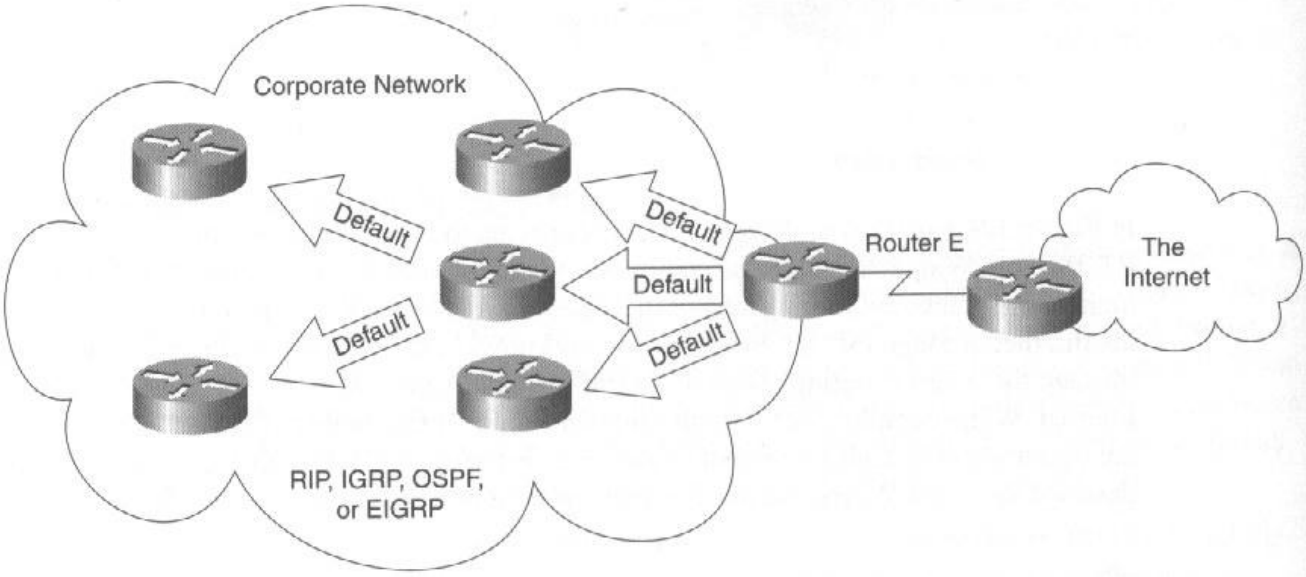
points to Router C as the next hop. Subsequently, Router C does not need to send Branch-RT any routing updates—Router C can have a passive interface or route filter on its link to Branch-RT, reducing overhead on the link between Branch-RT and Router C.

NOTE Router C still needs to know how to reach the branch office LAN, so Branch-RT needs to send a single route upstream to Router C.

Propagation of Default Routes

Usually, it is not enough to configure a default route on one router and be done with it. The default route must be shared with other routers, as depicted in Figure 3-10.

Figure 3-10 *Propagation of a Default Route*

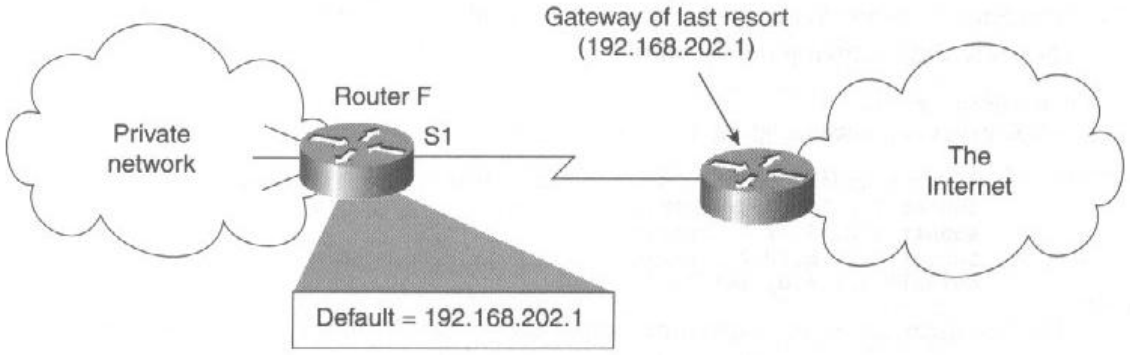


In Figure 3-10, Router E is the exit point for traffic leaving the corporate network and heading out to the Internet. Router E is configured with a default route so that it doesn't have to hold the entire Internet routing table. This satisfies Router E's requirement to reach points on the Internet, but what about the other routers in the corporate network? They must also learn about this default route or they will not know how to forward a packet destined to the Internet (they will drop the packet). To inform others of the default route, Router E propagates the default route with the interior routing protocol. In the same way that it carries normal routes, the routing protocol carries the default route to all parts of the corporate network. As each router receives the default route, it can then forward traffic destined for the Internet. This process is called *originating* (or *advertising*) *a default route*. Because Router E starts it all, E is the *originator*.

Originating a Default Route with RIP

It's easy, sometimes dangerously easy, to configure a router to originate a default route with RIP. All you need to do is configure the default route itself and RIP takes care of the rest. RIP automatically advertises the default out all RIP-enabled interfaces. Consider the example network depicted in Figure 3-11.

Figure 3-11 *RIP and Default Routing*



In the network in Figure 3-11, Router F has a default route to the Internet. The following is Router F's RIP configuration:

```
router rip
 network 172.18.0.0
!
ip route 0.0.0.0 0.0.0.0 192.168.202.1
```

The command **ip route 0.0.0.0 0.0.0.0 192.168.202.1** configures a default route in the router. It's a static route with a special network number **0.0.0.0** and mask **0.0.0.0**. The router with address **192.168.202.1** is called the *gateway of last resort*. The gateway of last resort is the target for all destinations Router F does not know how to reach (to which it does not have a route).

To verify that the default route is in effect, issue the **show ip route** command and take note of the following lines highlighted in boldface:

```
RTF#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default

Gateway of last resort is 192.168.202.1 to network 0.0.0.0

C    192.168.202.0 is directly connected, Serial1
     172.18.0.0 255.255.255.0 is subnetted, 5 subnets
C       172.18.10.0 is directly connected, Ethernet0
R       172.18.11.0 [120/2] via 172.18.10.1, 00:00:17, Ethernet0
```

```
R       172.18.12.0 [120/1] via 172.18.10.1, 00:00:17, Ethernet0
C       172.18.44.0 is directly connected, Serial0
R       172.18.1.0 [120/3] via 172.18.44.1, 00:00:02, Serial0
S*   0.0.0.0 0.0.0.0 [1/0] via 192.168.202.1
```

The line beginning with **Gateway of last resort** confirms that the default route is active and consistent with the configuration.

The last line of the output displays the default route itself. It has a code of **S***, meaning it is a static route and a default route (the asterisk signifies a default route).

Immediately after you configure the default route, RIP starts advertising it to other routers, as shown in the following debug output:

```
RTF#deb ip rip
RIP protocol debugging is on

RIP: sending update to 255.255.255.255 via Serial0 (172.18.44.2)
     subnet 172.18.10.0, metric 1
     subnet 172.18.11.0, metric 3
     subnet 172.18.12.0, metric 2
     default 0.0.0.0, metric 1
```

The boldface line in the preceding output shows that Router F is advertising the default route out its Serial0 interface. The result is a bit surprising because a default route is inherently a static route and you usually have to redistribute statics into RIP before they are advertised (see the use of the **redistribute static** command in "Resolving Issues with VLSM and Classful Routing Protocols" earlier in this chapter). But in this case, 0.0.0.0 is the exception: It is advertised by RIP even if **redistribute static** is missing.

NOTE Because originating a default route with RIP is so easy, care must be taken to avoid the accidental propagation of default routes. Someone might configure a default route on a RIP router, thinking that the default is for that router only and not realizing that his or her router is sending 0.0.0.0 to the rest of the network. When routers hear multiple default routes, the wrong one might be trusted (if it has a lower hop count). This leads to reachability problems.

Originating a Default Route with IGRP

As with RIP, you can configure a default route on an IGRP router like this:

```
router igrp
 network 172.18.0.0
!
ip route 0.0.0.0 0.0.0.0 192.168.202.1
```

The default route 0.0.0.0 is used locally by the router; however, it cannot be advertised to any other router with IGRP. IGRP does not propagate 0.0.0.0 even if you redistribute it into IGRP with the **redistribute static** command.

To make IGRP originate a default route and advertise it to others, configure IGRP as follows (this also works for EIGRP):

```
router igrp 100
 redistribute static
 network 172.18.0.0
!
ip route 30.0.0.0 255.0.0.0 192.168.202.1
ip default-network 30.0.0.0
```

where **ip route 30.0.0.0 255.0.0.0 192.168.202.1** creates a static route to a fictitious network 30.0.0.0 (a major net). This network does not actually exist in the organization. It's just a placeholder and a substitute for **ip route 0.0.0.0 0.0.0.0 192.168.202.1**.

The command **ip default-network 30.0.0.0** declares that 30.0.0.0, the fictitious network, is to be treated as a gateway of last resort. Traffic with an unknown destination is sent to this network. The router knows the route to this network is through 192.168.202.1 because of the previous command, **ip route 30.0.0.0 255.0.0.0 192.168.202.1**.

The command **redistribute static** is necessary to redistribute the static route, 30.0.0.0, into IGRP so that it can be advertised to other IGRP routers. Upon receiving 30.0.0.0, other routers will use it as a default route. This is equivalent to a RIP router receiving 0.0.0.0.

To verify the configuration of the originator of the default, issue **show ip route**:

```
IGRP-Router#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
       T - traffic engineered route

Gateway of last resort is 192.168.202.1 to network 30.0.0.0

S*   30.0.0.0/8 [1/0] via 192.168.202.1
     172.16.0.0/24 is subnetted, 2 subnets
C       172.16.13.0 is directly connected, Serial0
C       172.16.2.0 is directly connected, Serial1
<lines deleted for brevity>
```

The preceding output agrees with the configuration: **Gateway of last resort** is correct, and network 30.0.0.0 is coded with **S***, indicating it is a default route. If you forget to configure the **ip default-network** command, the asterisk (*) will not appear and the static will not be a default route.

Originating a Default Route with EIGRP

To originate a default route with EIGRP, you can use the same method described in "Originating a Default Route with IGRP." Alternatively, you can configure 0.0.0.0 and redistribute it into EIGRP like this:

```
router eigrp 100
 redistribute static
 network 172.18.0.0
 default-metric 1544 2000 255 1 1500
!
ip route 0.0.0.0 0.0.0.0 192.168.202.1
```

The command **ip route 0.0.0.0 0.0.0.0 192.168.202.1** creates a default route, with 192.168.202.1 as the gateway of last resort.

The command **redistribute static** redistributes 0.0.0.0 (and any other statics) into EIGRP so that it can be advertised to other routers.

The command **default-metric 1544 2000 255 1 1500** defines the starting EIGRP metric for the static routes that are redistributed into EIGRP. In order, the elements of the metric are:

- **Bandwidth**—In kilobits per second.
- **Delay**—In tens of microseconds.
- **Reliability**—A number from 0 to 255, where 255 is most reliable.
- **Loading**—A number from 0 to 255, where 255 is a fully loaded (100% saturated) link.
- **MTU (Maximum Transmission Unit)**—In bytes.

Originating a Default Route with OSPF

Like IGRP and EIGRP, OSPF does not automatically propagate a default route. To originate a default route with OSPF, configure OSPF with the **default-information originate** command:

```
router ospf 10
 network 172.18.43.0 0.0.0.255 area 0
 <other network commands>
 default-information originate metric 20 metric-type 1
!
ip route 0.0.0.0 0.0.0.0 172.18.44.2
```

where **ip route 0.0.0.0 0.0.0.0 172.18.44.2** configures the default route and gateway of last resort.

The command **default-information originate metric 20 metric-type 1** tells OSPF to propagate 0.0.0.0 with a starting metric of 20. This behaves like the **redistribute** command except it only redistributes the default route. The parameter **metric-type 1** tells OSPF to advertise the route as an external type 1 route. External type 1 and type 2 routes are covered in "Configuring Redistribution," earlier in this chapter.

When an OSPF router receives the default route from the originator, you will see 0.0.0.0 in the table, as shown in the following output of **show ip route**:

```
OSPF-Router#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
       T - traffic engineered route

Gateway of last resort is 172.18.43.2 to network 0.0.0.0

     172.18.0.0/16 is variably subnetted, 5 subnets, 2 masks
O E2    172.18.200.0/24 [110/10] via 172.18.43.2, 01:39:39, Serial1
O E2    172.18.44.0/24 [110/10] via 172.18.43.2, 01:44:42, Serial1
C       172.18.43.0/24 is directly connected, Serial1
C       172.18.4.0/22 is directly connected, Ethernet0
C       172.18.1.0/24 is directly connected, Serial0
O*E1 0.0.0.0/0 [110/45] via 172.18.43.2, 00:01:12, Serial1
```

In the preceding output, the two lines highlighted in boldface confirm that this router received the default route. It came from 172.18.43.2; therefore, 172.18.43.2 is this router's gateway of last resort.

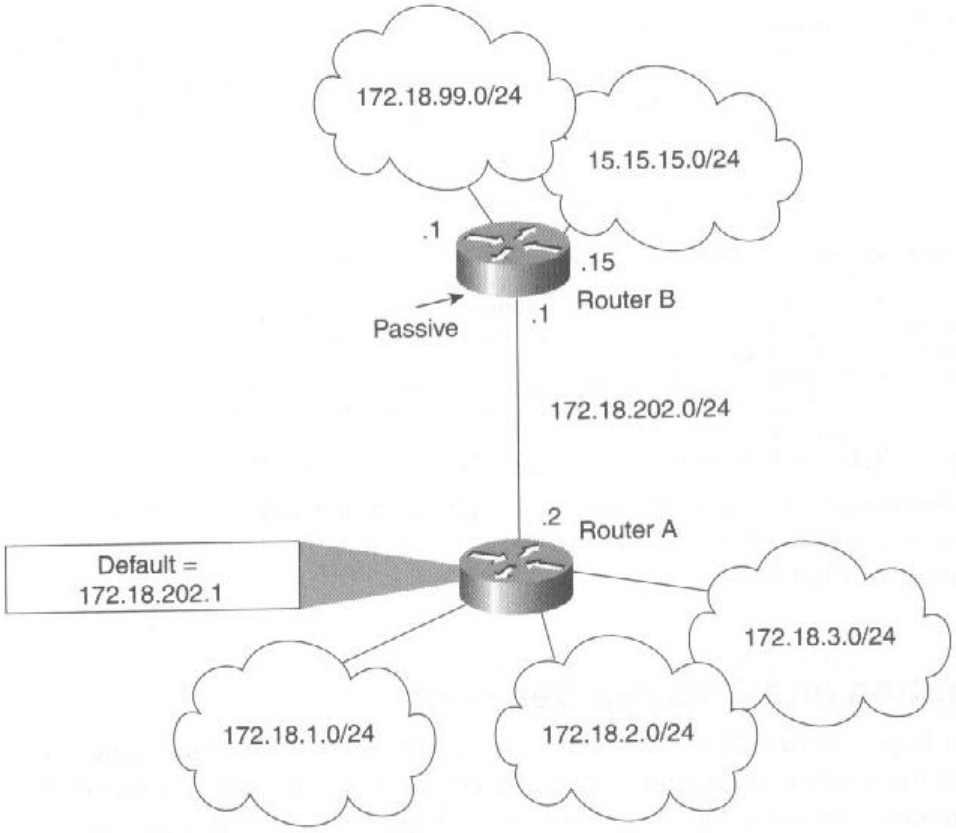
Default Routing and Classful Behavior

When you use a default route to reach a subnet of a connected major net, you must ensure that the router is configured with the **ip classless** global config command. Without **ip classless**, the router takes a classful posture and problems might surface. Consider the scenario depicted in Figure 3-12.

In the network pictured in Figure 3-12, Router A is classful (configured with **no ip classless**) and has a default route pointing to Router B as the gateway of last resort. Everything in Router A's routing table looks acceptable:

```
RTA#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default

Gateway of last resort is 172.18.202.1 to network 0.0.0.0

C    172.18.202.0 is directly connected, Serial0
     172.18.0.0/24 is subnetted, 3 subnets
C       172.18.1.0 is directly connected, Ethernet0
C       172.18.2.0 is directly connected, Ethernet1
C       172.18.3.0 is directly connected, Serial1
S*   0.0.0.0 0.0.0.0 [1/0] via 172.18.202.1
```

Figure 3-12 *Problems with Default Routing and Classful Behavior*



As shown in the preceding output, Router A lacks routes to 172.18.99.0/24 and 15.15.15.0/24, but it has a default route to Router B for all unknown destinations. Will the default route let Router A reach *both* 172.18.99.0/24 and 15.15.15.0/24? The answer is no, as shown by the following ping tests.

Issuing a ping to 15.15.15.15 gives:

```
RTA#ping 15.15.15.15
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 15.15.15.15, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms
```

As shown in the preceding output, Router A can reach 15.15.15.0/24 through the default route (exclamation points signify successful receipt of ping reply packets).

But issuing a ping to 172.18.99.1 gives:

```
RTA#ping 172.18.99.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.18.99.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)
```

The preceding output shows that Router A *cannot* reach 172.18.99.0/24 (dots indicate ping timeouts). So, what's the difference?

As a classful router, Router A checks for a direct connection to the major net before it forwards a packet. If the packet is destined to a directly connected major net (called a *local domain*), the router ignores the default route and looks in its routing table for a route to the destination. If a route does not exist, the packet is dropped.

Destination 172.18.99.1 is part of Router A's local domain. Subsequently, Router A does not use the default route and it drops the ping packets in the absence of a route to 172.18.99.0/24.

Packets to 15.15.15.0/24, on the other hand, are forwarded to the gateway of last resort because it is outside Router A's local domain and no routes exist for that destination.

To disable classful behavior (checking of the local domain), issue the **ip classless** global config command, like so:

```
RTA#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
RTA(config)#ip classless
```

The **ip classless** command toggles the router from classful mode to classless mode. With this command, the router has no notion of traditional classful addressing nor major nets. It simply looks at network prefixes. If Router A doesn't have a route to a destination, it uses the default route. The local domain check is skipped.

Issuing a ping to 172.18.99.1 with **ip classless** configured gives:

```
RTA#ping 172.18.99.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.18.99.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/4/8 ms
```

The preceding output shows that **ip classless** has resolved the problem of reaching the unknown destination 172.18.99.0/24.

NOTE An alternate way to resolve the problem with default routing and classful behavior is to configure a static route that points the major net to default route 0.0.0.0. Looking back to the initial problem in Figure 3-12, you can configure **ip route 172.18.0.0 255.255.0.0 0.0.0.0** in Router A. This allows classful Router A to reach unknown subnets of 172.18.0.0 through 0.0.0.0 (**no ip classless** is configured).

Configuring Route Summarization

As mentioned in Chapter 1, route summarization (also called aggregation) is the consolidation of multiple, contiguous routes into a single generalized route. It is recommended that you use summarization whenever you can, as your network addressing allows, to promote efficient and stable routing. If you are deploying a network from scratch, definitely plan your addressing so you can leverage summarization.

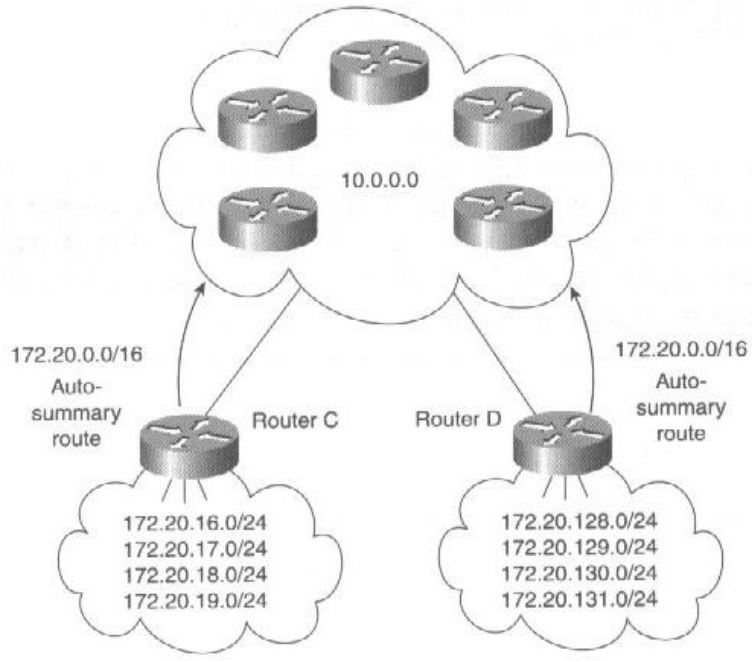
Summarization typically applies to classless routing protocols such as EIGRP and OSPF. RIP and IGRP are more primitive and do not have the summarization capabilities covered in this chapter.

NOTE BGP is a popular exterior routing protocol with rich summarization capabilities. Consult the Bibliography for sources of BGP information. Additionally, a good BGP primer is freely available at http://www.cisco.com/univercd/cc/td/doc/cisintwk/ics/index.htm on Cisco's Web site. The primer can also be found by searching the Cisco Web site for *internetworking case studies*.

Understanding EIGRP Auto-Summarization

By default, EIGRP automatically summarizes across major net boundaries. Often, this is nothing to worry about; however, when you deploy discontiguous subnets, as EIGRP supports, you might have to disable EIGRP auto-summarization. Consider the following scenario with auto-summarization enabled (see Figure 3-13). Assume EIGRP is the only routing protocol.

Figure 3-13 *EIGRP Auto-Summarization*



In the network depicted in Figure 3-13, major net 172.20.0.0 is broken up and separated by major net 10.0.0.0—in other words, the subnets of 172.20.0.0 are discontiguous. Router C thinks it's sitting on a major net boundary because it has direct connections to both 10.0.0.0 and 172.20.0.0.

With EIGRP auto-summarization enabled, Router C sends a summary route for all of 172.20.0.0 into 10.0.0.0 because it is a major net boundary. It has no need to tell the routers in 10.0.0.0 about the internal subnets of 172.20.0.0—or so it thinks.

NOTE Classful protocols RIP and IGRP also automatically summarize across major net boundaries. They are classful, so they naturally assume all subnets of a major net are contiguous. You cannot disable auto-summarization in RIP or IGRP.

After Router C sends 172.20.0.0/16 upstream, the details of the subnets beneath Router C are now lost. The summary route generalizes the four subnets with a single route.

Router D is also an EIGRP router and automatically summarizes its subnets into 10.0.0.0. Router D sends 172.20.0.0/16 into the 10.0.0.0 network. This is the same route advertised by Router C.

In this scenario, you can identify the following problems caused by EIGRP auto-summarization:

- Router D never learns the details of the subnets beneath Router C. Router C's summary route throws away the specific routes to 172.20.16.0/24, 172.20.17.0/24, 172.20.18.0/24, and 172.20.19.0/24.
- In the same way, Router C never learns the details of the subnets beneath Router D.
- Routers in the 10.0.0.0 cloud receive a summary route for 172.20.0.0 but from different directions. When they choose the route with the lowest metric, that route reaches the destinations beneath either Router C or Router D, but not both.

To solve the problems with auto-summarization and discontiguous subnets, disable auto-summarization with the EIGRP **no auto-summary** command. The following is Router C's configuration:

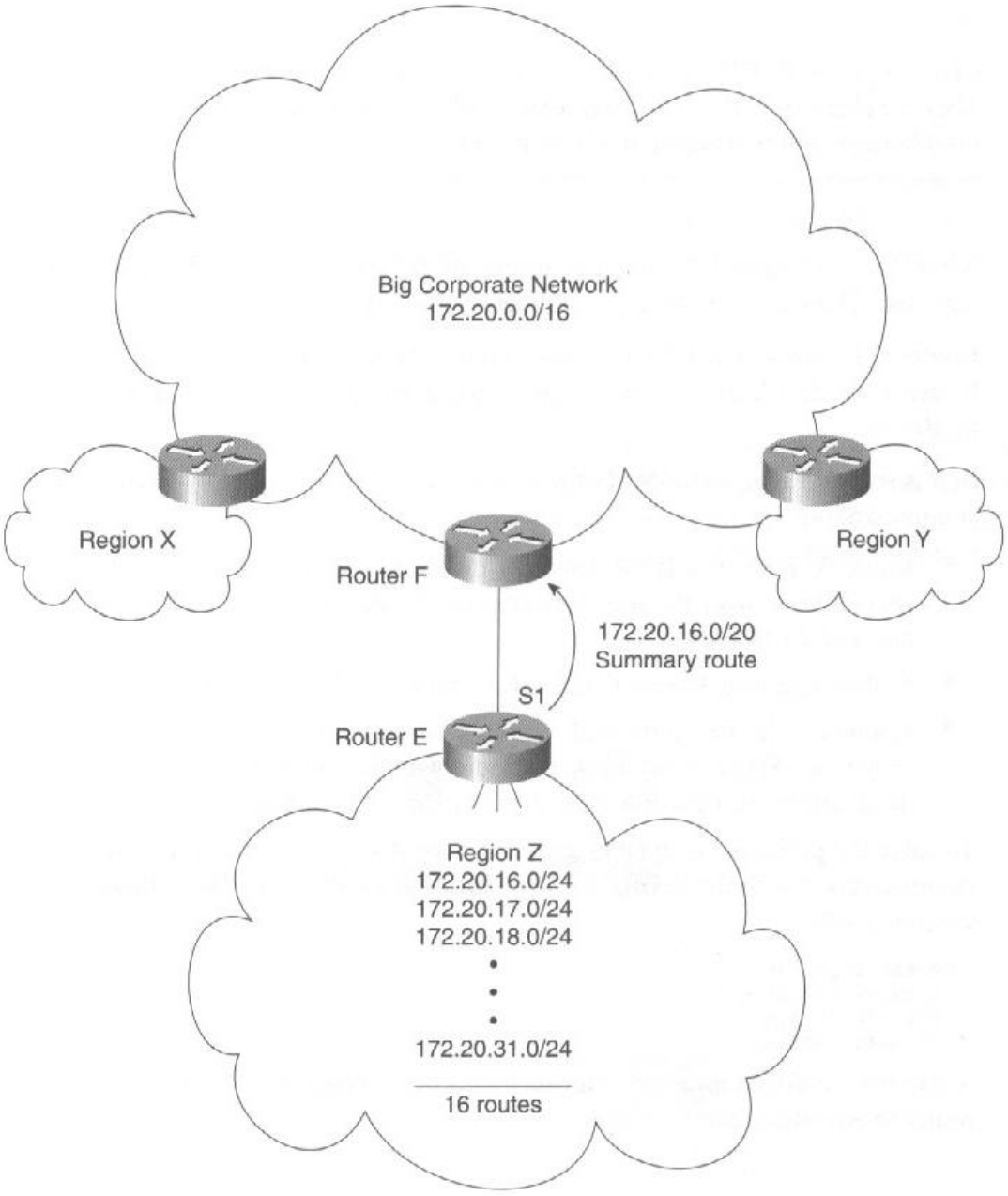
```
router eigrp 100
 network 172.20.0.0
 network 10.0.0.0
 no auto-summary
```

where the command **no auto-summary** disables EIGRP auto-summarization and forces the router to advertise specific routes.

Configuring EIGRP Summarization

EIGRP allows you to manually summarize multiple contiguous routes into a more simplified and generalized route. Summarization is key for network scalability, routing efficiency, and router resource conservation. Consider the application for route summarization shown in Figure 3-14.

Figure 3-14 *Applying EIGRP Route Summarization*



In Figure 3-14, Router E is responsible for advertising Region Z's routes to the big corporate network. Normally, Router E advertises each /24 route separately and injects 16 routes into the big corporate network—this is the default behavior.

Due to some good address planning, the routes in Region Z are contiguous. Subsequently, Router E can summarize and reduce the number of routes it advertises—that is, it can advertise a single route that covers all of Region Z's address space. The rest of the organization, such as Regions X and Y, can still reach everything in Region Z, and every router benefits from sending and maintaining fewer routes.

The following is Router E's configuration with summarization applied:

```
interface Serial1
 ip address 172.20.77.2 255.255.255.0
 ip summary-address eigrp 100 172.20.16.0 255.255.240.0
!
router eigrp 100
 network 172.20.0.0
 no auto-summary
```

where **ip summary-address eigrp 100 172.20.16.0 255.255.240.0** is the key command and summarizes all 16 routes inside Region Z with 172.20.16.0/20 (mask 255.255.240.0 equals /20). Interface Serial1 gets this command because it is the interface pointing to the big corporate network—the summary route exits out this interface. Keywords **eigrp 100** define the associated EIGRP autonomous system (necessary because a router might route for more than one autonomous system).

The command **no auto-summary** might not be necessary. If major net 172.20.0.0 has discontiguous subnets, the command is needed. See "Understanding EIGRP Auto-Summarization," earlier in this chapter.

NOTE EIGRP route summarization is performed at the interface level. Although OSPF has separate summarization commands for internal (area-to-area) and external (redistributed) routes, EIGRP summarizes both internal and external routes with **ip summary-address eigrp** at the interface.

To understand why 172.20.16.0/20 adequately describes all 16 routes in Region Z, examine the octet where the mask falls (the third octet). Expand the third octet (decimal 16) into binary and apply the /20 mask, as shown in Example 3-1.

Example 3-1 *Analysis of Summarization with 172.20.16.0/20*

```
Third octet = Decimal 16 = 0001-0000
Third octet of /20 mask  = 1111-0000
-------------------------------------
Route summarizes:        0001-xxxx
                         (where x is 0 or 1)
```

As shown in Example 3-1, the /20 mask sets the third octet to 0001-*xxxx*, where *x* can be a zero or a one. This means all possible combinations for the third octet are 0001-0000 through 0001-1111 (decimal 16 through 31). Putting the range of the third octet into the rest of the address yields an address range of 172.20.16.0 through 172.20.31.255 (the last octet can be anything from 0 to 255). This equals the address space of the subnets in Region Z; therefore, 172.20.16/20 is the correct summary route for Region Z.

A look at Router E's routing table shows that the summary route is installed:

```
RTE#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
       T - traffic engineered route

Gateway of last resort is not set

D       172.20.16.0/20 is a summary, 00:06:31, Null0
C       172.20.16.0/24 is directly connected, Serial0.1
C       172.20.17.0/24 is directly connected, Serial0.2
C       172.20.18.0/24 is directly connected, Serial0.3
C       172.20.19.0/24 is directly connected, Serial0.4
<lines deleted for brevity>
```

The entry shown in boldface in the preceding output is the EIGRP summary route for Region Z. Because this router is the originator, the summary route points to Null0.

Likewise, Router F receives and installs the summary route:

```
RTF#sh ip ro
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route, o - ODR
       T - traffic engineered route
```

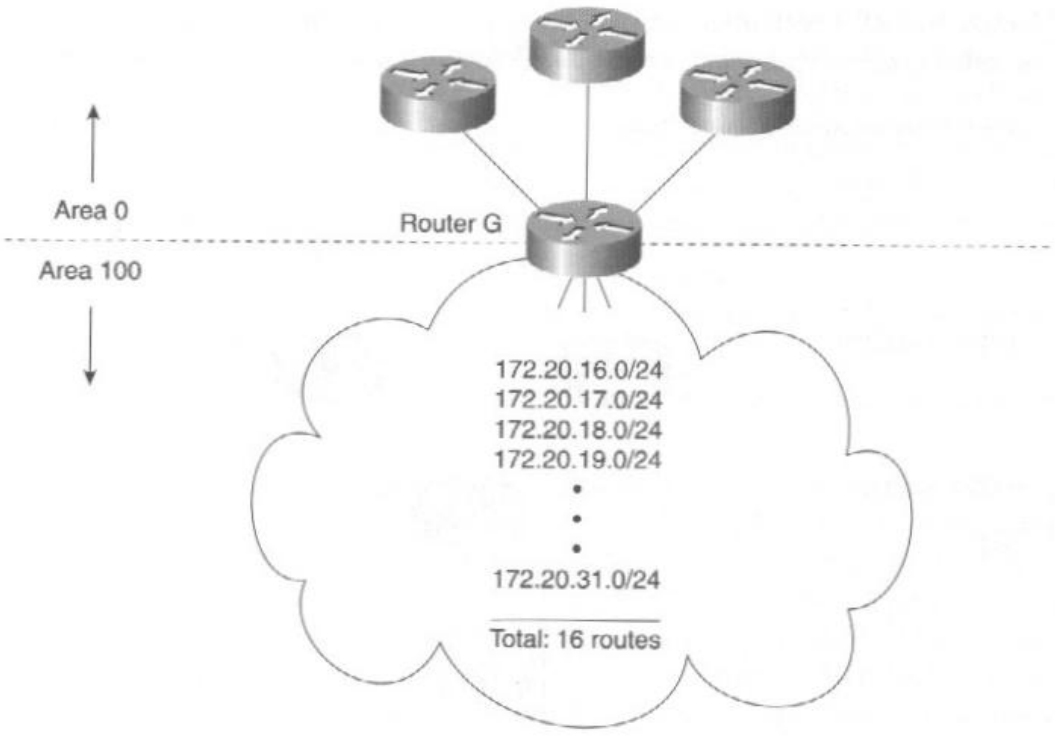
```
Gateway of last resort is not set

     172.20.0.0/16 is variably subnetted, 2 subnets, 2 masks
D       172.20.16.0/20 [90/1920000] via 172.20.77.2, 00:16:48, Serial1
C       172.20.77.0/30 is directly connected, Serial1
<lines deleted for brevity>
```

The boldface entry in the preceding output is the summary route sent to this router (Router F) by Router E.

Configuring OSPF Summarization Between Areas

OSPF also supports manual summarization of routes on area border routers (ABRs). Consider the following topology (Figure 3-15).

Figure 3-15 *Summarizing OSPF Routes Between Areas*



In Figure 3-15, Router G is an ABR between area 0 and area 100 and is responsible for advertising the 16 routes in area 100 to area 0 (the backbone area). Normally, it does not summarize but simply injects 16 routes into area 0. This is another opportunity to summarize because the 16 routes are contiguous. Again, good address planning makes this possible.

To summarize the routes in area 100 with a single summary route, configure Router G with the OSPF **area range** command, like this:

```
router ospf 25
 network 172.20.16.0 0.0.15.255 area 100
 network 172.20.0.0 0.0.3.255 area 0
 area 100 range 172.20.16.0 255.255.240.0
```

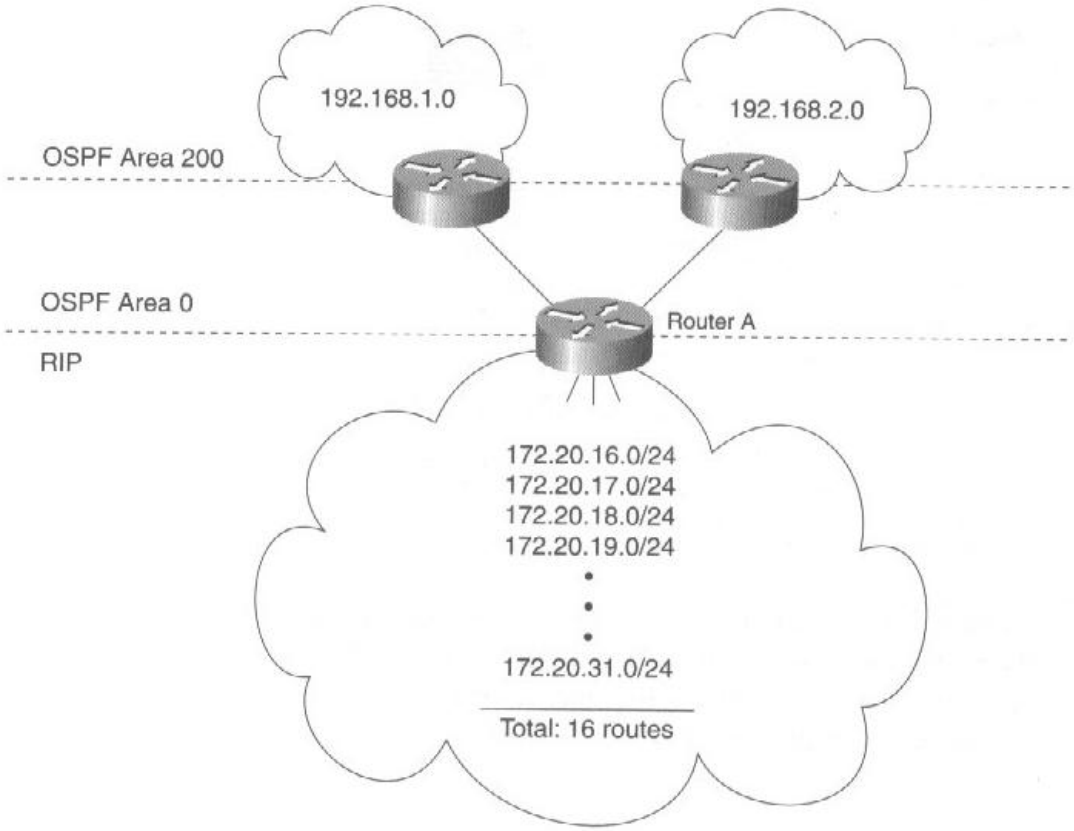
where **area 100 range 172.20.16.0 255.255.240.0** is the key command that summarizes the routes in area 100 with a single route 172.20.16.0/20. For an explanation of why 172.20.16.0/20 summarizes the 16 routes inside area 100, see the preceding section "Configuring EIGRP Summarization," which has the same summary route.

The other commands configure OSPF routing (see Chapter 2).

Configuring OSPF Summarization During Redistribution

Unlike EIGRP, OSPF maintains a separate process for summarizing redistributed routes. This is called *external route summarization*. Consider the scenario depicted in Figure 3-16.

Figure 3-16 *OSPF External Route Summarization*



Router A redistributes RIP into OSPF. Normally, the redistribution injects 16 routes, each with a /24 mask, into area 0. Instead of doing that, Router A can summarize the 16 routes and inject just a single route 172.20.16.0/20 into area 0. The following is Router A's OSPF configuration:

```
router ospf 25
 network 172.20.0.0 0.0.3.255 area 0
 summary-address 172.20.16.0 255.255.240.0
```

where **summary-address 172.20.16.0 255.255.240.0** is the key command that summarizes all 16 RIP routes into a single route 172.20.16.0/20. Router A then advertises this summary route to area 0 so that the other OSPF backbone routers learn the route to the RIP destinations.

Deploying Policy Routing with Route Maps

Policy routing enables you to direct traffic over user-defined paths based on the flexible syntax of access lists. With policy routing, you use enhanced filters called *route maps* to override normal forwarding decisions like those based on dynamic routing protocols. Route maps contain your criteria for identifying traffic and your instructions on how that traffic should be forwarded. You might want to do this to support certain routing policies, such as these:

- You want different applications (Web, e-mail, Telnet) to travel over different paths. That is, you want some applications to travel over normal paths determined by routing protocols, but you need other applications to travel over alternate paths—perhaps for performance or bandwidth allocation reasons.
- For legal, contractual, or security reasons, certain types of traffic must go over a different path than other types of traffic.
- You need to assign links to different groups of people for billing purposes. Each group uses and pays for its own bandwidth pipe.

In addition to directing traffic over different paths, policy routing enables you to set IP precedence values in packets. This marks (or *classifies*) packets with a certain quality of service (QoS) level that queuing and discarding services might use to prioritize traffic in the network. Another IOS service, Committed Access Rate (CAR), also classifies packets by setting IP precedence values. CAR and other advanced QoS features are covered in Chapter 5, "Deploying Advanced Quality of Service Features." Chapter 4, "Deploying Basic Quality of Service Features," covers IP precedence, QoS concepts, and basic QoS features.

This section covers both forms of policy routing: forwarding traffic over user-defined paths and classifying traffic with IP precedence.

NOTE This section requires familiarity with the access list syntax. Consult Chapter 6 as needed.

Forwarding Traffic with Route Maps

Route maps define criteria for matching packets and instructions for what to do with them. In the case of packet forwarding, the instructions define the next hop router to which the packet should be sent or the interface by which the packet should exit. This function is similar to static routing (see "Configuring a Static Route" in Appendix E), but with more control: You can control exactly which packets get forwarded and which do not by using the flexible syntax of access lists.

Route maps are built of one or more entry declarations that each contain so-called *match* and *set* statements. For example, the following commands configure a route map called TESTMAP:

```
route-map TESTMAP permit 20
 match ip address 100
 set ip next-hop 192.168.10.130
```

The command **route-map TESTMAP permit 20** creates a route map entry with a sequence of 20. A route map might contain multiple entries to support multiple policy-routing instructions. The sequence number identifies each entry so you can edit or delete entries without disturbing the rest of the route map. The keyword **permit** tells the router that packets matching the entry, as specified by **match** commands in the entry, should be processed by the instructions (**set** commands) found in the entry—this is the default behavior. If the entry is marked **deny** instead of **permit**, the packets matching the entry are *not* policy routed: They are sent back through the normal forwarding channel.

NOTE Space out your route map sequence numbers so you have the flexibility to insert new sequences between the entries as needed. When policy-routing a packet, the router compares the packet to each entry in the route map in order of sequence. The entry with the lowest sequence number is examined first.

The command **match ip address 100** defines the matching criteria of the entry. Here, access list 100—configured elsewhere in the router—defines the subset of packets that are policy-routed by the entry. Access lists are covered in Chapter 6.

The command **set ip next-hop 192.168.10.130** defines the action performed on packets that meet the criteria defined by the previous **match** statement. Here, the action is to send the matching packets to the next hop router, 192.168.10.130.

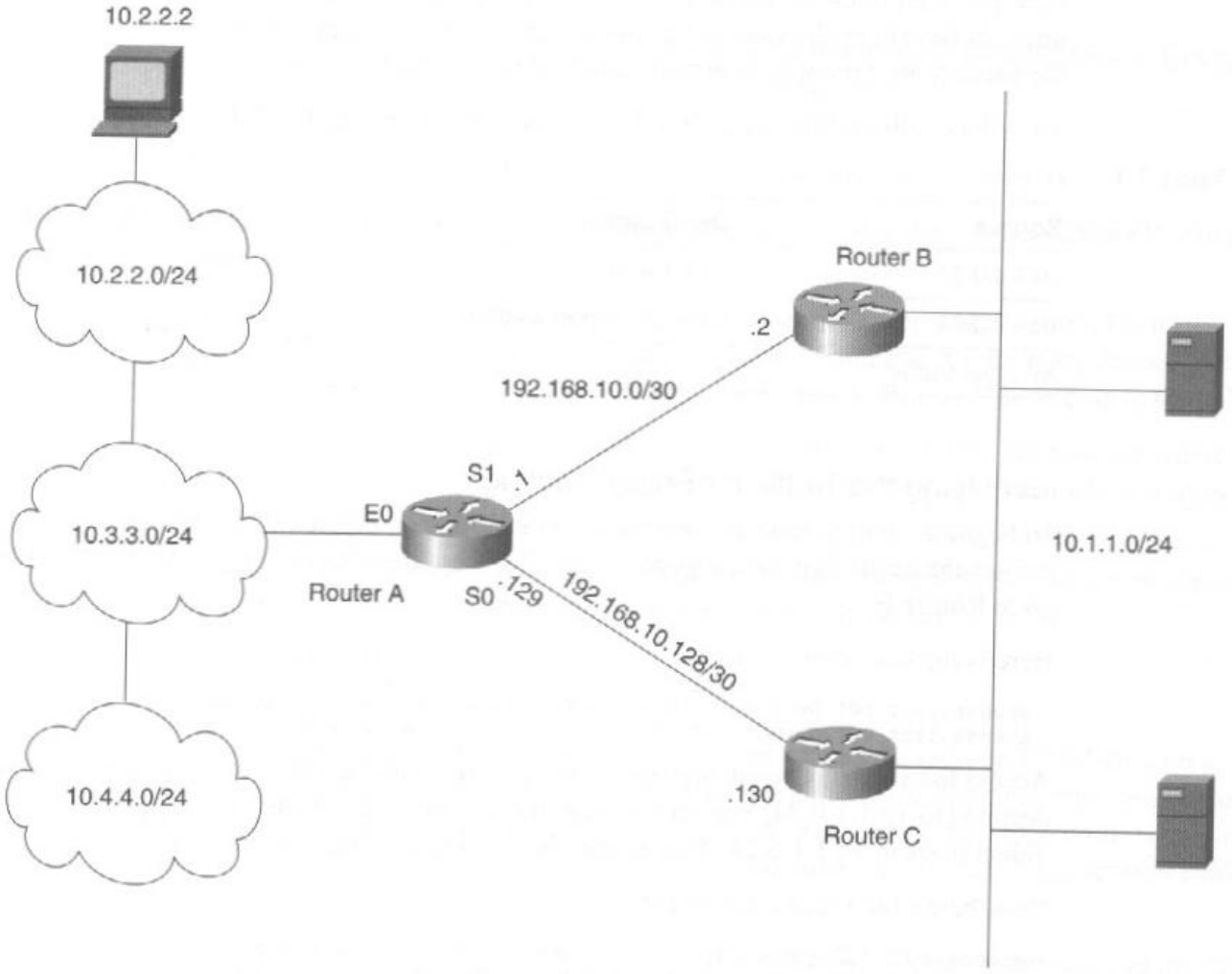
In the TESTMAP example, the first entry is given a sequence number of 20, enabling you to insert new entries before or after the initial entry. This is useful because packets are processed against a route map one entry at a time in order of sequence until a match is found. When a matching entry is found, the router processes the packet according to **set** commands in the entry and restarts the policy-routing procedure with the next packet.

NOTE A packet that does not match a route map entry is routed (forwarded) normally.

Policy Routing: An Example

To demonstrate the use and configuration of route maps, consider the scenario depicted in Figure 3-17.

Figure 3-17 *An Example for Policy Routing (Route Maps)*



Router A has two paths to subnet 10.1.1.0/24: one through Router B and another through Router C. Dynamic routing protocols such as OSPF, RIP, EIGRP, and the like might dictate to Router A that the preferred path to 10.1.1.0/24 is through either Router B or Router C, or both—*both* being equal-cost load balancing. The client 10.2.2.2 represents a node that is used for validating the policy-routing configuration (see "Validating Policy Routing Configuration," later in this chapter).

Now, suppose you must support the following policy on Router A:

- All traffic from subnet 10.4.4.0/24 to subnet 10.1.1.0/24 should travel over the link between Router A and Router C.
- Telnet sessions from clients in 10.2.2.0/24 to servers in 10.1.1.0/24 should also travel over the link between Router A and Router C.
- All other traffic passing through Router A to 10.1.1.0/24 should use the link between Router A and Router B.

This policy overrides the forwarding decisions determined by routing protocols. In this example, the policy covers all conceivable traffic from Router A to 10.1.1.0/24, but that does not have to be the case. When packets do not match the policy defined by a route map, the packets are forwarded normally as if the route map never existed.

To assist configuration, the policy for Router A is summarized in Table 3-2.

Table 3-1 *Policies to be Implemented on Router A in Figure 3-17*

Source	Destination	Next Hop Router
10.4.4.0/24	10.1.1.0/24	RTC
10.2.2.0/24	10.1.1.0/24 port 23 (Telnet)	RTC
All other traffic	10.1.1.0/24	RTB

Identifying the Traffic for Policy Routing

To begin the policy-routing configuration on Router A, define two access lists: one that defines the traffic that should go to Router C and another that defines the traffic that should go to Router B.

Here is the first access list:

```
access-list 101 permit ip 10.4.4.0 0.0.0.255 10.1.1.0 0.0.0.255
access-list 101 permit tcp 10.2.2.0 0.0.0.255 10.1.1.0 0.0.0.255 eq telnet
```

Access list 101 consists of two rules. The first rule matches all traffic from 10.4.4.0/24 destined to 10.1.1.0/24. The second rule matches all traffic from 10.2.2.0/24 to just the Telnet ports in 10.1.1.0/24. This access list will be used to direct traffic to Router C.

Now, here's the second access list:

```
access-list 102 permit ip 10.3.3.0 0.0.0.255 10.1.1.0 0.0.0.255
access-list 102 permit ip 10.2.2.0 0.0.0.255 10.1.1.0 0.0.0.255
```

Access list 102 consists of two rules, also. The first rule matches all traffic from 10.3.3.0/24 to 10.1.1.0/24. The second rule matches all traffic from 10.2.2.0/24 to 10.1.1.0/24. This access list will be used to direct traffic to Router B.

NOTE Chapter 6 covers access list fundamentals and configuration.

Access list 102 matches *all* traffic from 10.2.2.0/24 to 10.1.1.0/24, including Telnet packets that are also matched by access list 101. This means for the intended policy to work, the route map must be configured to process packets against access list 101 before access list 102.

Creating the Route Map Entries

Next, create Router A's route map. The first entry of the route map forwards traffic to Router C. This coincides with the desired policy. Here's the first entry:

```
route-map MYMAP permit 20
 match ip address 101
 set ip next-hop 192.168.10.130
```

The command **route-map MYMAP permit 20** creates a route map called MYMAP with an initial entry whose sequence is 20.

The command **match ip address 101** defines the matching criteria of the entry. Here, all packets that match access list 101 are subject to the **set** commands of the entry. Recall that access list 101 encompasses the packets that need to be sent to Router C according to the policy.

The command **set ip next-hop 192.168.10.130** defines the action taken when packets meet the **match** criteria. The action is to forward the packets to neighboring Router C, whose address is 192.168.10.130 (refer to Figure 3-17).

Now, configure a second entry to MYMAP. This entry matches packets that need to be directed to Router B and sets the next hop to 192.168.10.2, Router B's address.

```
route-map MYMAP permit 40
 match ip address 102
 set ip next-hop 192.168.10.2
```

The command **route-map MYMAP permit 40** creates a second entry in MYMAP with a sequence of 40. This entry has a higher sequence number than the previous entry, **route-map MYMAP permit 20**. This means packets are checked against sequence 40, only if they do not match sequence 20. As mentioned earlier, a packet that does not match any sequence is routed normally.

The command **match ip address 102** configures the matching criteria of this entry to the rules in access list 102. Recall that access list 102 encompasses the packets that need to be sent to Router B.

The command **set ip next-hop 192.168.10.2** defines the action taken when packets meet the criteria in the **match** statement. The action is to forward the packets to Router B whose address is 192.168.10.2 (refer to Figure 3-17).

Apply the Route Map to the Proper Interface

With the route map configuration complete, all that remains is to apply the route map to an interface. Route maps do not do anything until you apply them to an interface.

When you apply a route map to an interface, the interface you select is important. Route maps inspect and process packets as they enter the router; therefore, you must apply the route map to the interface that *receives* the traffic to be policy routed.

In the example depicted in Figure 3-17, the proper place to apply MYMAP is interface Ethernet0 because that is the interface on Router A that receives the traffic requiring policy routing.

The following commands apply MYMAP to Ethernet0 on Router A:

```
interface Ethernet0
 ip address 10.3.3.1 255.255.255.0
 ip policy route-map MYMAP
 ip route-cache policy
```

The command **ip policy route-map MYMAP** applies MYMAP to interface Ethernet0. All traffic that enters Router A through this interface is subject to the policies defined in MYMAP.

The command **ip route-cache policy** enables a feature called *fast-switched policy routing*. This feature, available in IOS 11.3 and later, substantially increases the performance of policy routing by caching information so the CPU doesn't have to process every policy-routed packet.

Validating Policy-Routing Configuration

According to the policy, in the example in Figure 3-17, Telnet packets from 10.2.2.0/24 to 10.1.1.0/24 should go over the link between Router A and Router C, but all other traffic from 10.2.2.0/24 to 10.1.1.0/24 should go over the link between Router A and Router B. To test that the route map is indeed making this happen, you can initiate a Telnet session from a client on 10.2.2.0/24 to a server on 10.1.1.0/24. This client is depicted in Figure 3-17 with the address 10.2.2.2.

After you successfully establish the Telnet session, use the command **show ip cache policy** to verify policy routing:

```
RTA#sh ip cac pol
Total adds 1, total deletes 0

Type Routemap/sequence      Age       Interface        Next Hop
NH   MYMAP/20               00:00:04  Serial0          192.168.10.130
```

The preceding output shows one entry in the policy-routing cache. The following list explains the data:

- **NH** stands for next hop. This means the route map is forwarding in response to the **set ip next-hop** command. The other possible keyword in the **Type** column is **Int**, which indicates forwarding by the **set interface** command (see "Other Policy-Routing Commands," later in this chapter).
- **MYMAP/20** is the name of the route map and the sequence number of the route map entry.
- **00:00:04** is the age of the cache entry. Four seconds ago, a packet matched the policy defined by MYMAP's sequence number 20 and caused the router to create an entry in the policy-routing cache. Only the first packet creates the entry and starts the timer. Subsequent packets that match MYMAP sequence 20 are cache hits and do not restart the timer.
- **Serial0** is the output interface for the policy-routed packets.
- **192.168.10.130** is the address of the next hop router (Router C).

NOTE The command **show ip cache policy** works only when you configure fast-switched policy routing with the **ip route-cache policy** interface command.

The preceding data is a good indication that policy routing is working as expected. An additional test is to have the client 10.2.2.2 send non-Telnet packets—for example, ping packets—to 10.1.1.0/24. According to the policy defined by MYMAP, such packets are forwarded to Router B (192.168.10.2).

After 10.2.2.2 issues the pings, check the policy-routing cache again:

```
RTA#sh ip cac pol
Total adds 2, total deletes 0

Type Routemap/sequence      Age       Interface        Next Hop
NH   MYMAP/20               00:18:32  Serial0          192.168.10.130
NH   MYMAP/40               00:00:27  Serial1          192.168.10.2
```

The last line in the preceding output indicates that the other entry in MYMAP, sequence 40, is actively forwarding packets to Router B, whose address is 192.168.10.2.

Another way of verifying policy routing is to watch the flow of packets through the router with packet-level debugging. This proves to be a more definitive test than checking the policy-routing cache but requires that you disable fast-switched policy routing.

Before you enable packet-level debugging with the **debug ip packet** command, you should create a simple access list that filters the debugging down to the specific source you are interested in watching. Otherwise, you might overload the router with too many debugging messages.

To watch packets sourced by 10.2.2.2, create an access list on Router A that permits only source address 10.2.2.2 (refer to Figure 3-17):

```
RTA#conf t
RTA(config)#access-list 10 permit 10.2.2.2
```

Next, disable fast-switched policy routing on the interface. This forces the router's CPU to examine every packet during policy routing so that debug messages per packet are displayed.

```
RTA(config)#int e0
RTA(config-if)#no ip route-cache policy
RTA(config-if)#end
```

Then, enable packet-level debugging with access list 10 as the filter:

```
RTA#deb ip packet detail 10
IP packet debugging is on (detailed) for access list 10
```

Have the client 10.2.2.2 initiate a Telnet connection to 10.1.1.1 and watch the router's debugging messages:

```
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial0), g=192.168.10.130, len 44, forward
    TCP src=1025, dst=23, seq=3268019581, ack=0, win=512 SYN
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial0), g=192.168.10.130, len 40, forward
    TCP src=1025, dst=23, seq=3268019582, ack=2048489389, win=32120 ACK
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial0), g=192.168.10.130, len 64, forward
    TCP src=1025, dst=23, seq=3268019582, ack=2048489389, win=32248 ACK PSH
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial0), g=192.168.10.130, len 52, forward
    TCP src=1025, dst=23, seq=3268019606, ack=2048489407, win=32248 ACK PSH
```

The preceding debugging output confirms that Router A is properly forwarding Telnet packets from 10.2.2.2 to 10.1.1.1. That is, policy routing is sending those packets to the next-hop router 192.168.10.130 (Router C). This agrees with the policy defined by entry 20 in MYMAP. The next-hop address is highlighted in boldface and has the output **g=192.168.10.130**. The output **dst=23** confirms that the packets belong to Telnet.

When the same client does a ping to 10.1.1.1, the following debugging messages appear:

```
RTA#
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial1), g=192.168.10.2, len 84, forward
    ICMP type=8, code=0
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial1), g=192.168.10.2, len 84, forward
    ICMP type=8, code=0
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial1), g=192.168.10.2, len 84, forward
    ICMP type=8, code=0
```

```
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial1), g=192.168.10.2, len 84, forward
    ICMP type=8, code=0
IP: s=10.2.2.2 (Ethernet0), d=10.1.1.1 (Serial1), g=192.168.10.2, len 84, forward
    ICMP type=8, code=0
```

The preceding output confirms that Router A is properly forwarding non-Telnet packets from 10.2.2.2 to 10.1.1.1. The output **g=192.168.10.2** (highlighted in boldface) indicates that the next-hop router is 192.168.10.2, Router B. This agrees with the policy defined by entry 40 in MYMAP. The output **ICMP** confirms that the packets are ping packets.

Classifying Packets with Route Maps

In addition to directing traffic over certain paths, policy routing can also classify or mark packets with a user-defined quality of service (QoS). This is accomplished with route maps that set the IP precedence value in packets as they enter the router. QoS mechanisms in the router and in the rest of the network can then read the precedence value of a packet and prioritize it relative to other packets. For a complete discussion on QoS and IP precedence, see Chapter 4.

NOTE Another IOS service, Committed Access Rate (CAR), also classifies packets by setting IP precedence values. CAR is covered in Chapter 5.

Route maps that classify traffic are just like the route maps covered previously in the "Forwarding Traffic with Route Maps" section except they implement the command **set ip precedence** instead of the command **set ip next-hop**. The following is an example:

```
interface Ethernet0
 ip address 10.2.2.1 255.255.255.0
 ip policy route-map QOSMAP
 ip route-cache policy
!
route-map QOSMAP permit 20
 match ip address 101
 set ip precedence critical
!
route-map QOSMAP permit 40
 match ip address 102
 set ip precedence flash
```

In the preceding configuration, the command **ip policy route-map QOSMAP** applies the route map called QOSMAP to interface Ethernet0. As mentioned previously, route maps inspect and apply policies to packets as they enter the router; therefore, route maps must be applied to the interface that receives the packets to be classified. In other words, route maps set precedence as packets enter the router.

The command **ip route-cache policy** enables fast-switched policy routing on the interface. See "Apply the Route Map to the Proper Interface" earlier in this chapter for more on fast-switched policy routing.

The command **route-map QOSMAP permit 20** defines the first entry in the route map named QOSMAP. The following are the subcommands that define the match-set criteria of this entry:

- **match ip address 101**: This is the matching criteria of entry 20. Packets that match access list 101 are subject to the **set** commands of the entry.
- **set ip precedence critical**: This defines the action taken when packets meet the **match** criteria. The action is to set the precedence value in the packet to **critical**, which is equivalent to decimal value 5.

Similarly, the command **route-map QOSMAP permit 40** defines the second and final entry in the route map. In this entry, the matching criteria are defined by access list 102 and the action is to set the precedence of the matching packets to **flash** (decimal value 3).

When you issue the **set ip precedence** command, you can specify precedence values by name or by decimal value—if you specify the decimal value, IOS automatically converts it to the name when you view the configuration. To see a mapping of names to decimal values, use the context-level help feature built into IOS, like this:

```
RTA(config-route-map)#set ip precedence ?
  <0-7>           Precedence value
  critical        Set critical precedence (5)
  flash           Set flash precedence (3)
  flash-override  Set flash override precedence (4)
  immediate       Set immediate precedence (2)
  internet        Set internetwork control precedence (6)
  network         Set network control precedence (7)
  priority        Set priority precedence (1)
  routine         Set routine precedence (0)
```

The preceding output displays a list of precedence levels by name, with their decimal values shown in parentheses.

NOTE It is highly recommended that you avoid using precedence levels 6 and 7. These are reserved for network control packets such as those belonging to routing protocols. This leaves you with six user-definable precedence levels, 0 through 5.

Setting Next-Hop and Precedence in Tandem

If needed, you can create route map entries that combine the two **set** operations, **set ip next-hop** and **set ip precedence**. The following is an example:

```
route-map COMBO permit 20
 match ip address 103
 set ip precedence critical
 set ip next-hop 192.168.10.130
```

The preceding entry for route map COMBO applies two **set** operations to the packets matching access list 103: The router sets the precedence to **critical** (decimal 5) and then forwards the packets to next-hop router 192.168.10.130.

Other Policy-Routing Commands

The following is a list of other match and set statements for policy routing:

- **match length** *min max*—Matches the length (in bytes) of the packet instead of using an access list. The parameters *min* and *max* specify the minimum and maximum length criteria.
- **set interface** *type number* [*...type number*]—Tells the router where to output the packets meeting the matching criteria. Instead of specifying a next-hop address, this specifies an exit interface on the router (**Serial0**, for example).
- **set default interface** *type number* [*...type number*]—Routes to the interface specified by this **set** command only if there is no explicit route for the destination address in the routing table. Use this to provide an alternate default route for packets meeting the match criteria.
- **set ip default next-hop** *ip-address* [*...ip-address*]—Routes to the next-hop address specified by this **set** command only if there is no explicit route for the destination address in the routing table. Use this to provide an alternate default route for packets meeting the match criteria.
- **set ip tos** *type-of-service*—Sets the type of service (ToS) bits instead of IP precedence.

TIP When using the **match length** command, ensure that routing protocol packets are not accidentally policy routed to the wrong neighboring routers. Otherwise, routing adjacency problems might occur.

By default, packets that are generated by the router are not policy routed. To change this, issue the global command **ip local policy route-map**, like this:

```
RTA(config)#ip local policy route-map TESTMAP
```

where the keyword **TESTMAP** in the preceding command identifies the route map to use for policy routing locally generated packets.

Summary

As you extend, enhance, and optimize the capabilities of your network, you must deal with the crucial function of routing. This chapter covered some important routing services that help you manage network scalability, complexity, and stability. These services (and your mastery of them) directly impact the number of users and locations you can support, your ability to integrate new networks and topologies, and the stability of the network as it expands.

The following are the key concepts of this chapter:

- A passive interface is an interface on which you deliberately suppress the advertising of routing updates. Use the **passive-interface** command to make an interface passive.
- Route filters give you granular control over the routes sent and received by your router. To filter routes, use access lists and the **distribute-list** router config mode command.
- Redistribution is a service that allows the sharing of routes across different routing domains—for example, between RIP and OSPF. It transfers routes between distinct domains and makes connectivity possible across them. Enable this service with the **redistribute** router config mode command.
- When redistributing routes into OSPF, don't forget to use the **subnets** keyword if you need to redistribute both major nets and their subnets.
- The **administrative distance** is a priority assigned to a route based on its routing protocol. A route with a lower administrative distance is preferred over a route with a higher administrative distance.
- To prevent route feedback caused by mutual redistribution, you should always control redistribution with route filters.
- Although classful routing protocols do not support VLSM, sometimes you are forced to redistribute from a classless domain into a classful one, and routes might be lost in the process. As a workaround, you might be able to originate and redistribute static routes that are agreeable to the classful domain.
- A **default route** is a special route that tells the router how to reach unknown destinations. How you originate and propagate a default route depends on the routing protocol.
- Be careful when you configure a default route on a RIP router. RIP automatically advertises the default route out all RIP-enabled interfaces. Watch out for accidental propagation of default routes on the network.

- When you use a default route to reach a subnet of a connected major net, you must ensure that the router is configured with the **ip classless** global config command.
- EIGRP and OSPF support route summarization: the consolidation of multiple contiguous routes into a single generalized route to promote efficient and stable routing.
- EIGRP automatically summarizes across major net boundaries. When you deploy discontiguous subnets, you might have to disable EIGRP auto-summarization.
- Use the **ip summary-address** interface command to configure EIGRP route summarization.
- To summarize OSPF routes between areas, use the **area range** router config mode command. To summarize routes as they are redistributed into OSPF, use the **summary-address** router config mode command.
- Policy routing allows you to override normal forwarding decisions with route maps. Route maps are built of one or more entries that are created with the **route-map** global config command. Each entry contains **match** and **set** statements that define the routing policy.
- You can also use policy routing to set IP precedence and classify packets for QoS.