This chapter covers the following subjects:

- The Roots of EIGRP: An Overview of IGRP
- From IGRP to EIGRP
- Operation of EIGRP
- Configuring EIGRP
- Troubleshooting EIGRP

CHAPTER 7

# Enhanced Interior Gateway Routing Protocol (EIGRP)

First released in IOS 9.21, Enhanced Interior Gateway Routing protocol (EIGRP) is, as the name says, an enhancement of the Cisco Interior Gateway Routing Protocol (IGRP). The name is apt because unlike RIPv2, EIGRP is far more than the same protocol with some added extensions. Like IGRP, EIGRP is a distance vector protocol and uses the same composite metrics as IGRP uses. Beyond that, there are few similarities.

IGRP was discontinued as of IOS releases 12.2(13)T and 12.2(Rls4)S. Although innovative in its time, modern network operators who desire more capability than RIP provides move to EIGRP or OSPF, not the moderately more capable IGRP. However, to understand EIGRP, it is useful to first examine the protocol from which it evolved.

## The Roots of EIGRP: An Overview of IGRP

Cisco developed IGRP in the mid-1980s as an answer to the limitations of RIP, the most significant of which are the hop count metric and the 15-hop network size. IGRP calculated a composite metric from a variety of route variables and provided “knobs” for weighting the variables to reflect the specific characteristics and needs of the network. Although hop count is not one of these variables, IGRP did track hop count, and could be implemented on networks of up to 255 hops in diameter.

The other advantages that IGRP presented over RIP are

- Unequal-cost load sharing
- An update period three times longer than RIP’s
- A more efficient update packet format

The chief disadvantage of both IGRP and EIGRP is that they are proprietary to Cisco and therefore limited to Cisco products, whereas RIP is a part of any IP routing process on any platform.

The Cisco objective when developing IGRP was to create a versatile, robust protocol capable of being adapted to a variety of routed protocol suites. In addition to routing IP, IGRP could also route the ISO Connectionless Network Protocol (CLNP). This multiprotocol capability was also adapted in EIGRP, which can route not only IP but also IPX and AppleTalk.

From a high-altitude view, IGRP shares many operational characteristics with RIP. It is a classful distance vector protocol that periodically broadcasts its entire routing table—with the exception of routes suppressed by split horizon—to all its neighbors. Like RIP, IGRP broadcasts a request packet out all IGRP-enabled interfaces upon startup and performs a sanity check on received updates to verify that the source address of the packet belongs to the same subnet on which the update was received.[1] New update entries with reachable metrics are placed in the routing table, and an entry replaces an older entry to the same destination only if the metric is smaller. Split horizon with poisoned reverse, triggered updates, and holddown timers are used for stability; IGRP summarizes addresses at network boundaries.

Unlike RIP, which is accessed via UDP, the IGRP process is accessed directly from the IP layer as protocol 9.

## Process Domains

IGRP also uses the concept of process domains. By defining and tracking multiple process domains, you can isolate the communications within one domain from the communications within another domain. Traffic between the domains can then be closely regulated by redistribution (Chapter 11, "Route Redistribution") and route filtering (Chapter 13, "Route Filtering").

Figure 7-1 illustrates the contrast between process domains and routing domains. Here two autonomous systems (AS) are defined: AS 10 and AS 40. These systems are routing domains—a set of routers running one or more IGPs under a common administration. They communicate via an Exterior Gateway Protocol (in this case, Border Gateway Protocol, or BGP).

Within AS 10 are two IGRP process domains: IGRP 20 and IGRP 30. Under IGRP, the 20 and 30 are defined as autonomous system numbers. In this context, the numbers serve to distinguish two routing processes within the same routing domain. IGRP 20 and IGRP 30 communicate via the single router connected to both domains. This router runs both IGRP processes and automatically redistributes between them.

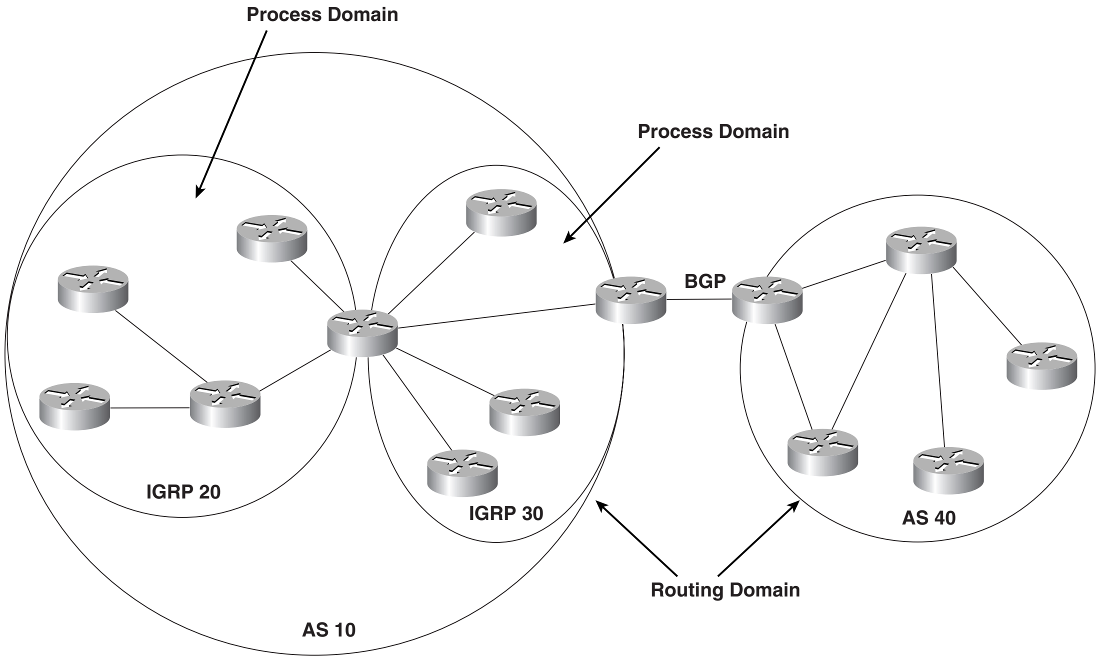
Within its updates, IGRP classifies route entries into one of three categories: interior routes, system routes, and exterior routes.

An *interior* route is a path to a subnet of the network address of the data link on which the update is being broadcast. In other words, a subnet advertised as an interior route is "local" to the major network to which the advertising router and the receiving router are commonly connected.

A *system* route is a path to a network address, which has been summarized by a network boundary router.

[1] This sanity check can be disabled with the command **no validate-update-source**.

**Figure 7-1** *An autonomous system number may specify a routing domain, which is a group of routers running one or more IGP processes under a single administrative domain. Under IGRP, an autonomous system number may also specify a process domain, which is a group of routers sharing routing information by means of a single routing process.*



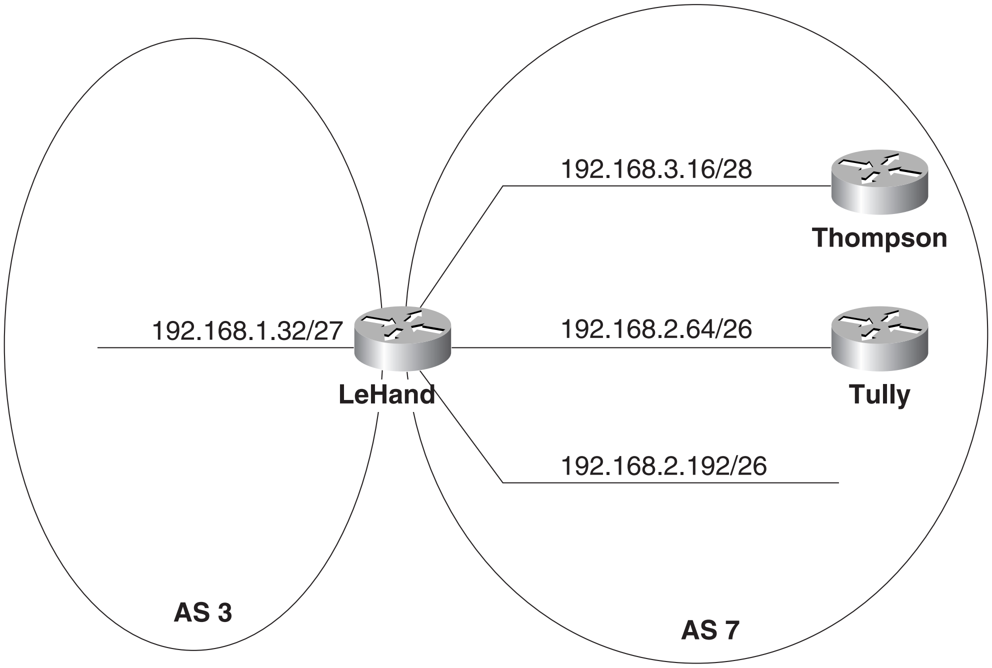
An *exterior* route is a path to a network that has been flagged as a *default network*. A default network is an address to which a router will send any packet that cannot be matched to a more specific destination.[2] Default networks and their configuration are covered in Chapter 12, "Default Routes and On-Demand Routing."

Figure 7-2 shows how IGRP uses these three categories. The routers LeHand and Tully are connected to subnet 192.168.2.64/26, so major network 192.168.2.0 is considered the "local" network shared by those two routers. LeHand is also attached to 192.168.2.192/26, which is another subnet of the network connecting the two routers. Therefore, LeHand advertises that subnet to Tully as an internal route.

However, the local network for LeHand and Thompson is 192.168.3.0. LeHand is the boundary router between major networks 192.168.2.0 and 192.168.3.0, so 192.168.2.0 will be advertised to Thompson as a system route. Likewise, 192.168.3.0 is advertised to Tully as a system route.

[2] Classifying a default network as an external route is unique to IGRP and EIGRP. Open protocols such as RIP and OSPF advertise default networks with the address 0.0.0.0.

**Figure 7-2** *LeHand advertises subnet 192.168.2.192/26 to Tully as an internal route. Network 192.168.3.0 is advertised to Tully as a system route, and 192.168.1.0 is advertised as an external route.*



192.168.1.0 is a network in another autonomous system, and LeHand has been configured to advertise that network address as a default route. 192.168.1.0 will therefore be advertised to both Thompson and Tully as an external route.

## IGRP Timers and Stability Features

The IGRP update period is 90 seconds. A random jitter variable of up to 20 percent is subtracted from each update time to prevent update timer synchronization, so the time elapsed between individual updates will vary from 72 to 90 seconds.

When a route is first learned, the invalid timer for that route is set for 270 seconds, or three times the update period. The flush timer is set for 630 seconds—seven times the update period. Each time an update is received for the route, these timers are reinitialized. If the invalid timer expires before an update is heard, the route is marked as unreachable. It will be held in the routing table and advertised as unreachable until the flush timer expires, at which time the route will be deleted from the table.

The 90-second timer used by IGRP, in comparison to the 30-second timer used by RIP, means that compared to RIP, IGRP uses less bandwidth for periodic updates. However, the trade-off is that in some cases IGRP might be slower to converge than RIP. For example, if a router goes offline, IGRP takes three times as long as RIP to detect the dead neighbor.

If a destination becomes unreachable or if the next-hop router increases the metric of a destination enough to cause a triggered update, the route will be placed in holddown for 280 seconds (three update periods plus 10 seconds). Until the holddown timer expires, no new information will be accepted about this destination. IGRP holddown may be disabled with the command **no metric holddown**. In loop-free topologies, where holddown has no real benefit, disabling the function can reduce reconvergence time.

The default timers can be changed with the following command:

```
timers basic update invalid holddown flush [sleeptime]
```

This command is also used to manipulate RIP timers with the exception of the *sleeptime* option. Sleeptime is a timer used to specify a period, in milliseconds, to delay a regular routing update after receiving a triggered update.

## IGRP Metrics

One of the most significant changes from RIP that IGRP introduces, and which carries over into EIGRP, is the use of multiple metric parameters, based on link characteristics. It is useful to study how IGRP handles these metrics to understand how EIGRP handles its metrics.

The link characteristics from which IGRP calculates its composite metric are bandwidth, delay, load, and reliability. By default, IGRP chooses a route based on bandwidth and delay. If a data link is thought of as a pipe, then bandwidth is the width of the pipe and delay is the length of the pipe. In other words, bandwidth is a measure of the carrying capacity, and delay is a measure of the end-to-end travel time. Load and reliability are taken into consideration only if the router is configured to do so. IGRP also tracks the smallest Maximum Transmission Unit (MTU) along each route, although the MTU is not used in the composite metric calculation. The quantities associated with IGRP's composite metric on a specific interface can be observed with the **show interfaces** command (Example 7-1).

**Example 7-1** *The output of every show interface command includes metric statistics for the interface. This Ethernet interface shows MTU = 1500 bytes, bandwidth = 10 megabits per second, delay = 1000 microseconds, reliability = 100 percent, and load = 39 percent (minimum load).*

```
Newfoundland#show interfaces ethernet 0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c76.5b7c (bia 0000.0c76.5b7c)
  Internet address is 10.2.1.2/24
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:07, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
```

*continues*

**Example 7-1** *The output of every show interface command includes metric statistics for the interface. This Ethernet interface shows MTU = 1500 bytes, bandwidth = 10 megabits per second, delay = 1000 microseconds, reliability = 100 percent, and load = 39 percent (minimum load). (Continued)*

```
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     601753 packets input, 113607697 bytes, 0 no buffer
     Received 601753 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 input packets with dribble condition detected
     693494 packets output, 557731861 bytes, 0 underruns
     0 output errors, 5 collisions, 13 interface resets
     0 babbles, 0 late collision, 48 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
Newfoundland#
```

Bandwidth is expressed in units of kilobits per second. It is a static number used for metric calculation only and does not necessarily reflect the actual bandwidth of the link—that is, bandwidth is not measured dynamically. For example, the default bandwidth of a serial interface is 1544, whether the interface is attached to a T1 or a 56K line. This bandwidth number may be changed from the default with the **bandwidth** command.

IGRP updates use a three-octet number, referred to in this book as $BW_{IGRP}$, which is the inverse of the bandwidth scaled by a factor of $10^7$. So if the bandwidth of an interface is 1544, then

$$BW_{IGRP} = 10^7/1544 = 6476, \text{ or } 0x00194C.$$

Delay, like bandwidth, is a static figure and is not measured dynamically. It is displayed by the **show interface** command as DLY, in units of microseconds. The default delay of an interface may be changed with the **delay** command, which specifies the delay in tens of microseconds. Example 7-2 shows the **bandwidth** and **delay** commands used to change the defaults of the interface of Example 7-1.

**Example 7-2** *The bandwidth and delay commands are used to change the metric defaults of the e0 interface. The new quantities can be seen in the output of the* **show interfaces** *command.*

```
Newfoundland(config)#interface e0
Newfoundland(config-if)#bandwidth 75000
Newfoundland(config-if)#delay 5
Newfoundland(config-if)#^Z
Newfoundland#
%SYS-5-CONFIG_I: Configured from console by console
Newfoundland#show interfaces ethernet0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c76.5b7c (bia 0000.0c76.5b7c)
  Internet address is 10.2.1.2/24
```

**Example 7-2** *The bandwidth and delay commands are used to change the metric defaults of the e0 interface. The new quantities can be seen in the output of the* **show interfaces** *command. (Continued)*

```
  MTU 1500 bytes, BW 75000 Kbit, DLY 50 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:02, output 00:00:02, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     601888 packets input, 113637882 bytes, 0 no buffer
     Received 601888 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 input packets with dribble condition detected
     693646 packets output, 557884632 bytes, 0 underruns
     0 output errors, 5 collisions, 13 interface resets
     0 babbles, 0 late collision, 48 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
Newfoundland#
```

When carried in an IGRP update, delay is a three-octet number expressed in the same 10-microsecond units as specified by the **delay** command. To avoid confusion, this number will be referred to as $DLY_{IGRP}$, to differentiate it from DLY, in microseconds, observed with **show interface**. For example, if DLY is 50, then

$$DLY_{IGRP} = DLY/10 = 50/10 = 5, \text{ or } 0x000005.$$

IGRP also uses delay to indicate an unreachable route by setting $DLY_{IGRP}$ = 0xFFFFFF. This number translates to approximately 167.8 seconds, so the maximum end-to-end delay of an IGRP route is 167 seconds.

Because IGRP uses bandwidth and delay as its default metrics, these quantities must be configured correctly and consistently on all interfaces of all IGRP routers.

Changing the bandwidth or delay of an interface should be done only for good reasons and only with a full understanding of the results of those changes. In most cases, it is best to leave the default values unchanged. A notable exception is serial interfaces. As noted earlier in this section, serial interfaces on Cisco routers have a default bandwidth of 1544 no matter what the bandwidth is of the connected link. The **bandwidth** command should be used to set the interface to the actual bandwidth of the serial link.

It is important to note that OSPF also uses the bandwidth statement to calculate its metric. Therefore, if IGRP (or EIGRP) metrics are to be manipulated in a network where OSPF is also running, use the **delay** to influence IGRP. Changing the bandwidth will affect both IGRP and OSPF.

Table 7-1 lists the bandwidths and delays for a few common interfaces. (The default bandwidth of a serial interface is always 1544; Table 7-1 shows the figures that would result from using the **bandwidth** command to reflect the actual connected bandwidth.)

**Table 7-1** *Common $BW_{IGRP}$ and $DLY_{IGRP}$ quantities.*

| Media | Bandwidth | $BW_{IGRP}$ | Delay | $DLY_{IGRP}$ |
|---|---|---|---|---|
| 100M ATM | 100000K | 100 | 100μS | 10 |
| Fast Ethernet | 100000K | 100 | 100μS | 10 |
| FDDI | 100000K | 100 | 100μS | 10 |
| HSSI | 45045K | 222 | 20000μS | 2000 |
| 16M Token Ring | 16000K | 625 | 630μS | 63 |
| Ethernet | 10000K | 1000 | 1000μS | 100 |
| T1 | 1544K | 6476 | 20000μS | 2000 |
| DS0 | 64K | 156250 | 20000μS | 2000 |
| 56K | 56K | 178571 | 20000μS | 2000 |
| Tunnel | 9K | 1111111 | 500000μS | 50000 |

Reliability is measured dynamically and is expressed as an eight-bit number, where 255 is a 100 percent reliable link and 1 is a minimally reliable link. In the output of **show interface**, reliability is shown as a fraction of 255, for example, 234/255 (see Example 7-3).

**Example 7-3** *This interface shows a reliability of 234/255, or 91.8 percent.*

```
Casablanca#show interface ethernet0
Ethernet0 is up, line protocol is up
  Hardware is Lance, address is 0000.0c76.5b7c (bia 0000.0c76.5b7c)
  Internet address is 172.20.1.1 255.255.255.0
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec, rely 234/255, load 1/255
  Encapsulation ARPA, loopback not set, keepalive set (10 sec)
  ARP type: ARPA, ARP Timeout 4:00:00
  Last input 0:00:28, output 0:00:06, output hang never
  Last clearing of "show interface" counters 0:06:05
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     22 packets input, 3758 bytes, 0 no buffer
     Received 21 broadcasts, 0 runts, 0 giants
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     0 input packets with dribble condition detected
     125 packets output, 11254 bytes, 0 underruns
     39 output errors, 694 collisions, 0 interface resets, 0 restarts
     0 output buffer failures, 0 output buffers swapped out
Casablanca#
```

Load, in an IGRP update, is an eight-bit number. Load is represented in the output of **show interface** as a fraction of 255, such as 40/255 (Example 7-4); 1 is a minimally loaded link, and 255 is a 100 percent loaded link.

**Example 7-4** *This interface shows a load of 40/255, or 15.7 percent.*

```
Yalta#show interface serial 1
Serial1 is up, line protocol is up
  Hardware is HD64570
  Internet address is 172.20.20.2 255.255.255.0
  MTU 1500 bytes, BW 56 Kbit, DLY 20000 usec, rely 255/255, load 40/255
  Encapsulation HDLC, loopback not set, keepalive set (10 sec)
  Last input 0:00:08, output 0:00:00, output hang never
  Last clearing of "show interface" counters 0:05:05
  Output queue 0/40, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 10000 bits/sec, 1 packets/sec
  5 minute output rate 9000 bits/sec, 1 packets/sec
     456 packets input, 397463 bytes, 0 no buffer
     Received 70 broadcasts, 0 runts, 0 giants
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     428 packets output, 395862 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets, 0 restarts
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

If reliability or load is to be used as a metric or as part of a composite metric, the algorithm for calculating the metric must not allow sudden changes in the error rate or channel occupancy to destabilize the network. As an example, if a “raw,” or instantaneous, measure of load is used, a burst of heavy traffic could cause a route to go into holddown and an abrupt drop in traffic could trigger an update. To prevent frequent metric changes, reliability and load are calculated based on an exponentially weighted average with a five-minute time constant, which is updated every five seconds.

The composite metric for each IGRP route is calculated as

$$\text{metric} = [k1*BW_{IGRP(min)} + (k2*BW_{IGRP(min)})/(256\text{-}LOAD) + k3*DLY_{IGRP(sum)}] \times [k5/(RELIABILITY + k4)],$$

where $BW_{IGRP(min)}$ is the minimum $BW_{IGRP}$ of all the outgoing interfaces along the route to the destination and $DLY_{IGRP(sum)}$ is the total $DLY_{IGRP}$ of the route.

The values k1 through k5 are configurable weights; their default values are k1=k3=1 and k2=k4=k5=0. These defaults can be changed with the command:[3]

```
metric weights tos k1 k2 k3 k4 k5[3]
```

[3] *tos* is a relic of the Cisco original intention to have IGRP do type of service routing; this plan was never adopted, and *tos* in this command is always set to zero.

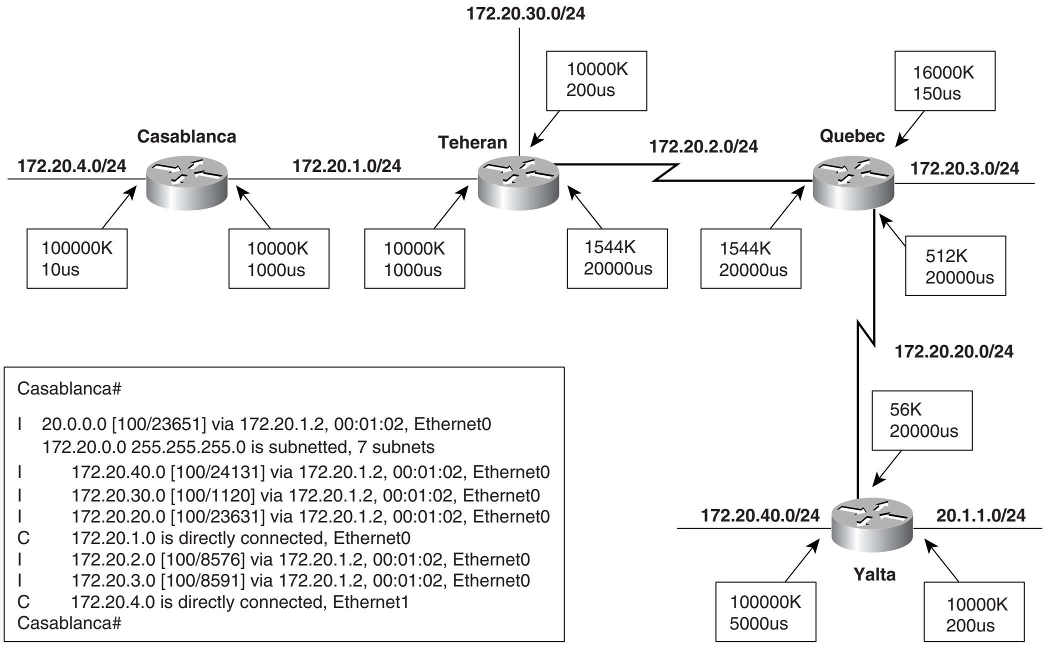
If k5 is set to zero, the [k5/(RELIABILITY+k4)] term is not used.

Given the default values for k1 through k5, the composite metric calculation used by IGRP reduces to the default metric:

$$\text{metric} = \text{BW}_{\text{IGRP(min)}} + \text{DLY}_{\text{IGRP(sum)}}$$

The network example in Figure 7-3 shows the bandwidths and delays configured on each interface and a forwarding database from one of the routers with the derived IGRP metrics.[4]

**Figure 7-3** *By default, the total delay is added to the minimum bandwidth to derive the IGRP metric.*



```
Casablanca#
I   20.0.0.0 [100/23651] via 172.20.1.2, 00:01:02, Ethernet0
    172.20.0.0 255.255.255.0 is subnetted, 7 subnets
I       172.20.40.0 [100/24131] via 172.20.1.2, 00:01:02, Ethernet0
I       172.20.30.0 [100/1120] via 172.20.1.2, 00:01:02, Ethernet0
I       172.20.20.0 [100/23631] via 172.20.1.2, 00:01:02, Ethernet0
C       172.20.1.0 is directly connected, Ethernet0
I       172.20.2.0 [100/8576] via 172.20.1.2, 00:01:02, Ethernet0
I       172.20.3.0 [100/8591] via 172.20.1.2, 00:01:02, Ethernet0
C       172.20.4.0 is directly connected, Ethernet1
Casablanca#
```

The routing table itself shows only the derived metric, but the actual variables recorded by IGRP for each route can be seen by using the command **show ip route** *address*, as in Example 7-5. Here the minimum bandwidth on the route from Casablanca to subnet 172.20.40.0/24 is 512K, at Quebec. The total delay of the route is (1000 + 20000 + 20000 + 5000) = 46000 microseconds:

$$\begin{aligned}
&\text{BW}_{\text{IGRP(min)}} = 10^7/512 = 19531\\
&\text{DLY}_{\text{IGRP(sum)}} = 46000/10 = 4600\\
&\text{metric} = \text{BW}_{\text{IGRP(min)}} + \text{DLY}_{\text{IGRP(sum)}} = 19531 + 4600 = 24131
\end{aligned}$$

[4] Also notice the administrative distance, which is 100 for IGRP.

**Example 7-5** *The metric for the route from Casablanca to subnet 172.20.40.0 is calculated from the minimum bandwidth of 512K and the total delay of 46000 microseconds.*

```
Casablanca#show ip route 172.20.40.0
Routing entry for 172.20.40.0 255.255.255.0
  Known via "igrp 1", distance 100, metric 24131
  Redistributing via igrp 1
  Advertised by igrp 1 (self originated)
  Last update from 172.20.1.2 on Ethernet0, 00:00:54 ago
  Routing Descriptor Blocks:
  * 172.20.1.2, from 172.20.1.2, 00:00:54 ago, via Ethernet0
      Route metric is 24131, traffic share count is 1
      Total delay is 46000 microseconds, minimum bandwidth is 512 Kbit
      Reliability 255/255, minimum MTU 1500 bytes
      Loading 1/255, Hops 2
```

Example 7-5 shows that IGRP also records the smallest MTU along the route in addition to the hop count. MTU is not used in the metric calculation. Hop count is the hop count reported by the next-hop router and is used only to limit the diameter of the network. By default, the maximum hop count is 100 and can be configured from 1 to 255 with the command **metric maximum-hops**. If the maximum hop count is exceeded, the route will be marked unreachable by setting the delay to 0xFFFFFF.

Note that all metrics are calculated from the outgoing interfaces along the route. For example, the metric for the route from Yalta to subnet 172.20.4.0/24 is different from the metric for the route from Casablanca to subnet 172.20.40.0/24. This is due to the differences in the configured bandwidth on the link between Yalta and Quebec and to the differences in the delay on the outgoing interfaces to the two destination subnets.

# From IGRP to EIGRP

The original motivation for developing EIGRP was simply to make IGRP classless. But early in the development the engineers working on the project recalled some academic proposals for a new kind of convergence algorithm and decided to use that algorithm in their extension of IGRP. The result was a protocol that, while retaining some concepts introduced with IGRP such as multiple metrics, protocol domains, and unequal-cost load balancing, is distinctly different from IGRP.

EIGRP is occasionally described as a distance vector protocol that acts like a link-state protocol. To recap the extensive discussion in Chapter 4, “Dynamic Routing Protocols,” a distance vector protocol shares everything it knows, but only with directly connected neighbors. Link-state protocols announce information only about their directly connected links, but they share the information with all routers in their routing domain or area.

All the distance vector protocols discussed so far run some variant of the Bellman-Ford (or Ford-Fulkerson) algorithm. These protocols are prone to routing loops and counting to infinity. As a result, they must implement loop-avoidance measures such as split horizon, route poisoning, and hold-down timers. Because each router must run the routing algorithm on received routes before passing those routes along to its neighbors, larger networks might be slow to converge. More important, distance vector protocols advertise routes; the change of a critical link might mean the advertisement of many changed routes.

Compared to distance vector protocols, link-state protocols are far less susceptible to routing loops and bad routing information. The forwarding of link-state packets is not dependent on performing the route calculations first, so large networks might converge faster. And only links or prefixes and their states are advertised, not routes, which means the change of a link will not cause the advertisement of all routes using that link.

Regardless of whether other routing protocols perform route calculations before sending distance vector updates to neighbors or after building a topological database, their common denominator is that they perform the calculations individually. In contrast to the Bellman-Ford algorithms used by most other distance vector protocols, EIGRP uses a system of *diffusing computations*—route calculations that are performed in a coordinated fashion among multiple routers—to attain fast convergence while remaining loop-free at every instant.

Although EIGRP updates are still vectors of distances transmitted to directly connected neighbors, they are nonperiodic, partial, and bounded. *Nonperiodic* means that updates are not sent at regular intervals; rather, updates are sent only when a metric or topology change occurs. *Partial* means that the updates will include only routes that have changed, not every entry in the route table. *Bounded* means that the updates are sent only to affected routers. These characteristics mean that EIGRP uses much less bandwidth than typical distance vector protocols use. This feature can be especially important on low-bandwidth, high-cost Wide Area Network (WAN) links.

Another concern when routing over low-bandwidth WAN links is the maximum amount of bandwidth used during periods of convergence, when routing traffic is high. By default, EIGRP uses no more than 50 percent of the bandwidth of a link. Later IOS releases allow this percentage to be changed with the command **ip bandwidth-percent eigrp**.

EIGRP is a classless protocol (that is, each route entry in an update includes a subnet mask). Variable-length subnet masks may be used with EIGRP not only for sub-subnetting as described in Chapter 6, "RIPv2, RIPng, and Classless Routing," but also for address aggregation—the summarization of a group of major network addresses.

EIGRP packets can be authenticated using an MD5 cryptographic checksum. The basics of authentication and MD5 are covered in Chapter 6; an example of configuring EIGRP authentication is included in this chapter.

Finally, a major feature of EIGRP is that it can route not only IP but also IPX and AppleTalk.

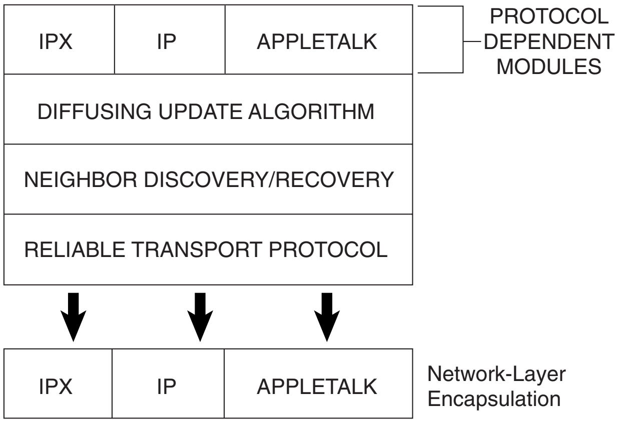
# Operation of EIGRP

EIGRP uses the same formula that IGRP uses to calculate its composite metric. However, EIGRP scales the metric components by 256 to achieve a finer metric granularity. So if the minimum configured bandwidth on the path to a destination is 512K and the total configured delay is 46000 microseconds, IGRP would calculate a composite metric of 24131. EIGRP, however, will multiply the bandwidth and delay components by 256 for a metric of 256 × 24131 = 6177536.

EIGRP has four components (Figure 7-4):

- Protocol-Dependent Modules
- Reliable Transport Protocol (RTP)
- Neighbor Discovery/Recovery
- Diffusing Update Algorithm (DUAL)

**Figure 7-4** *The four major components of EIGRP. RTP and neighbor discovery are lower-level protocols that enable the correct operation of DUAL. DUAL can perform route computations for multiple routed protocols.*


IPX
IP
APPLETALK
PROTOCOL DEPENDENT MODULES
DIFFUSING UPDATE ALGORITHM
NEIGHBOR DISCOVERY/RECOVERY
RELIABLE TRANSPORT PROTOCOL
IPX
IP
APPLETALK
Network-Layer Encapsulation

This section examines each EIGRP component, with particular emphasis on DUAL, and ends with a discussion of address aggregation.

## Protocol-Dependent Modules

EIGRP implements modules for IP, IPX, and AppleTalk, which are responsible for the protocol-specific routing tasks. For example, the IPX EIGRP module is responsible for exchanging route information about IPX networks with other IPX EIGRP processes and for passing the information to the DUAL. Additionally, the IPX module will send and receive SAP information.

As Figure 7-4 shows, the traffic for the individual modules is encapsulated within their respective network layer protocols. EIGRP for IPX, for example, is carried in IPX packets.

EIGRP automatically redistributes with other protocols in many cases:

- IPX EIGRP automatically redistributes with IPX RIP and NLSP.
- AppleTalk EIGRP automatically redistributes with AppleTalk RTMP.
- IP EIGRP automatically redistributes routes with IGRP if the IGRP process is in the same autonomous system.

Redistribution with other IP routing protocols is the subject of Chapter 11.

Configuration of EIGRP for IPX and AppleTalk is outside the scope of this book. Refer to the *Cisco Configuration Guide* for more information.

## Reliable Transport Protocol

The Reliable Transport Protocol (RTP) manages the delivery and reception of EIGRP packets. *Reliable delivery* means that delivery is guaranteed and that packets will be delivered in order.

Guaranteed delivery is accomplished by means of a Cisco-proprietary algorithm known as *reliable multicast*, using the reserved class D address 224.0.0.10. Each neighbor receiving a reliably multicast packet unicasts an acknowledgment.

Ordered delivery is ensured by including two sequence numbers in the packet. Each packet includes a sequence number assigned by the sending router. This sequence number is incremented by one each time the router sends a new packet. In addition, the sending router places in the packet the sequence number of the last packet received from the destination router.

In some cases, RTP may use *unreliable delivery*. No acknowledgment is required, and no sequence number will be included for unreliably delivered EIGRP packets.

EIGRP uses multiple packet types, all of which are identified by protocol number 88 in the IP header:

- **Hellos** are used by the neighbor discovery and recovery process. Hello packets are multicast and use unreliable delivery.
- **Acknowledgments** (ACKs) are Hello packets with no data in them. ACKs are always unicast and use unreliable delivery.
- **Updates** convey route information. Unlike RIP and IGRP updates, these packets are transmitted only when necessary, contain only necessary information, and are sent only to routers that require the information. When updates are required by a specific router, they are unicast. When updates are required by multiple routers, such as

upon a metric or topology change, they are multicast. Updates always use reliable delivery.

- **Queries and Replies** are used by the DUAL finite state machine to manage its diffusing computations. Queries can be multicast or unicast, and replies are always unicast. Both queries and replies use reliable delivery.
- **Requests** were a type of packet originally intended for use in route servers. This application was never implemented, and request packets are noted here only because they are mentioned in some older EIGRP documentation.

If any packet is reliably multicast and an ACK is not received from a neighbor, the packet will be retransmitted as a unicast to that unresponding neighbor. If an ACK is not received after 16 of these unicast retransmissions, the neighbor will be declared dead.

The time to wait for an ACK before switching from multicast to unicast is specified by the *multicast flow timer*. The time between the subsequent unicasts is specified by the *retransmission timeout* (RTO). Both the multicast flow timer and the RTO are calculated for each neighbor from the *smooth round-trip time* (SRTT). The SRTT is the average elapsed time, measured in milliseconds, between the transmission of a packet to the neighbor and the receipt of an acknowledgment. The formulas for calculating the exact values of the SRTT, the RTO, and the multicast flow timer are proprietary.

The following two subsections discuss the EIGRP components that use the various packet types.

## Neighbor Discovery/Recovery

Because EIGRP updates are nonperiodic, it is especially important to have a process whereby neighbors—EIGRP-speaking routers on directly connected networks—are discovered and tracked. On most networks, Hellos are multicast every five seconds, minus a small random time to prevent synchronization. On multipoint X.25, Frame Relay, and ATM interfaces, with access link speeds of T1 or slower, Hellos are unicast every 60 seconds.[5] This longer Hello interval is also the default for ATM SVCs and for ISDN PRI interfaces. In all cases, the Hellos are unacknowledged. The default Hello interval can be changed on a per interface basis with the command **ip hello-interval eigrp**.

When a router receives a Hello packet from a neighbor, the packet will include a *hold time*. The hold time tells the router the maximum time it should wait to receive subsequent Hellos. If the hold timer expires before a Hello is received, the neighbor is declared unreachable and DUAL is informed of the loss of a neighbor. By default, the hold time is three times the Hello interval—180 seconds for low-speed nonbroadcast multiaccess (NBMA) networks and 15 seconds for all other networks. The default can be changed on a per interface basis with the command **ip hold-time eigrp**. The capability to detect a lost

[5] Point-to-point subinterfaces send Hellos every 5 seconds.

neighbor within 15 seconds, as opposed to 180 seconds for RIP and 270 seconds for IGRP, is one factor contributing to EIGRP's fast reconvergence.

Information about each neighbor is recorded in a neighbor table. As Example 7-6 shows, the *neighbor table* records the IP address of the neighbor and the interface on which the neighbor's Hellos are received. The hold time advertised by the neighbor is recorded, as is the SRTT and the *uptime*—the time since the neighbor was added to the table. The RTO is the time, in milliseconds, that the router will wait for an acknowledgment of a unicast packet sent after a multicast has failed. If an EIGRP update, query, or reply is sent, a copy of the packet will be queued. If the RTO expires before an ACK is received, another copy of the queued packet is sent. The Q Count indicates the number of enqueued packets. The sequence number of the last update, query, or reply packet received from the neighbor is also recorded in the neighbor table. The RTP tracks these sequence numbers to ensure that packets from the neighbor are not received out of order. Finally, the H column records the order in which the neighbors were learned.

**Example 7-6** *The command* **show ip eigrp neighbors** *is used to observe the IP EIGRP neighbor table.*

```
Wright#show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address   Interface   Hold  Uptime    SRTT   RTO   Q   Seq
                          (sec)           (ms)        Cnt  Num
3   10.1.1.2  Et0           10  09:01:27    12   200   0   5
2   10.1.4.2  Se1           13  09:02:11    23   200   0   11
1   10.1.2.2  Et1           14  09:02:12     8   200   0   15
0   10.1.3.2  Se0           12  09:02:12    21   200   0   13
Wright#
```

## Diffusing Update Algorithm

DUAL is a convergence algorithm that replaces the Bellman-Ford or Ford-Fulkerson algorithms used by other distance vector protocols. The design philosophy behind DUAL is that even temporary routing loops are detrimental to the performance of a network. DUAL uses diffusing computations, first proposed by E. W. Dijkstra and C. S. Scholten,[6] to perform distributed shortest-path routing while maintaining freedom from loops at every instant. Although many researchers have contributed to the development of DUAL, the most prominent work is that of J. J. Garcia-Luna-Aceves.[7]

[6] Edsger W. Dijkstra and C. S. Scholten. "Termination Detection for Diffusing Computations." Information Processing Letters, Vol. 11, No. 1, pp. 1–4: 29 August 1980.

[7] J. J. Garcia-Luna-Aceves. "A Unified Approach for Loop-Free Routing Using Link States or Distance Vectors," ACM SIGCOMM Computer Communications Review, Vol. 19, No. 4, pp. 212–223: September 1989.
J. J. Garcia-Luna-Aceves. "Loop-Free Routing Using Diffusing Computations," IEEE/ACM Transactions on Networking, Vol. 1, No. 1, February 1993.

## DUAL: Preliminary Concepts

For DUAL to operate correctly, a lower-level protocol must ensure that the following conditions are met:[8]

- A node detects within a finite time the existence of a few neighbor or the loss of connectivity with a neighbor.
- All messages transmitted over an operational link are received correctly and in the proper sequence within a finite time.
- All messages, changes in the cost of a link, link failures, and new-neighbor notifications are processed one at a time within a finite time and in the order in which they are detected.

The Cisco EIGRP uses Neighbor Discovery/Recovery and RTP to establish these preconditions.

Before the operation of DUAL can be examined, a few terms and concepts must be described.

Upon startup, a router uses Hellos to discover neighbors and to identify itself to neighbors. When a neighbor is discovered, EIGRP will attempt to form an adjacency with that neighbor. An *adjacency* is a logical association between two neighbors over which route information is exchanged. When adjacencies have been established, the router will receive updates from its neighbors. The updates will contain all routes known by the sending routers and the metrics of those routes. For each route, the router will calculate a distance based on the distance advertised by the neighbor and the cost of the link to that neighbor.

The lowest calculated metric to each destination will become the *feasible distance* (FD) of that destination. For example, a router may be informed of three different routes to subnet 172.16.5.0 and may calculate metrics of 380672, 12381440, and 660868 for the three routes. 380672 will become the FD because it is the lowest calculated distance.

The *feasibility condition* (FC) is a condition that is met if a neighbor's advertised distance to a destination is lower than the router's FD to that same destination.

If a neighbor's advertised distance to a destination meets the FC, the neighbor becomes a *feasible successor*[9] for that destination. For example, if the FD to subnet 172.16.5.0 is 380672 and a neighbor advertises a route to that subnet with a distance of 355072, the neighbor will become a feasible successor; if the neighbor advertises a distance of 380928, it will not satisfy the FC and will not become a feasible successor.

The concepts of feasible successors and the FC are central to loop avoidance. Because feasible successors are always "downstream," (that is, a shorter metric distance to the

[8] J. J. Garcia-Luna-Aceves. "Area-Based, Loop-Free Internet Routing." Proceedings of IEEE INFOCOMM 94. Toronto, Ontario, Canada, June 1994.

[9] Successor simply means a router that is one hop closer to a destination—in other words, a next-hop router.

destination than the FD) a router will never choose a path that will lead back through itself, creating a loop. Such a path would have a distance larger than the FD.

Every destination for which one or more feasible successors exist will be recorded in a *topological table*, along with the following items:

- The destination's FD
- All feasible successors
- Each feasible successor's advertised distance to the destination
- The locally calculated distance to the destination via each feasible successor, based on the feasible successor's advertised distance and the cost of the link to that successor
- The interface connected to the network on which each feasible successor is found[10]

For every destination listed in the topological table, the route with the lowest metric is chosen and placed into the route table. The neighbor advertising that route becomes the *successor*, or the next-hop router to which packets for that destination are sent.

An example will help clarify these terms, but first a brief discussion of the network used in the examples in this section is necessary. Figure 7-5 shows the EIGRP-based network that is used throughout this and the next three subsections.[11] The command **metric weights 0 0 0 1 0 0** has been added to the EIGRP process so that only delay is used in the metric calculations. The **delay** command has been used with the numbers shown at each link; for example, the interfaces of routers Wright and Langley, connected to subnet 10.1.3.0, have been configured with a delay of 2. These steps have been taken to simplify the examples that follow.

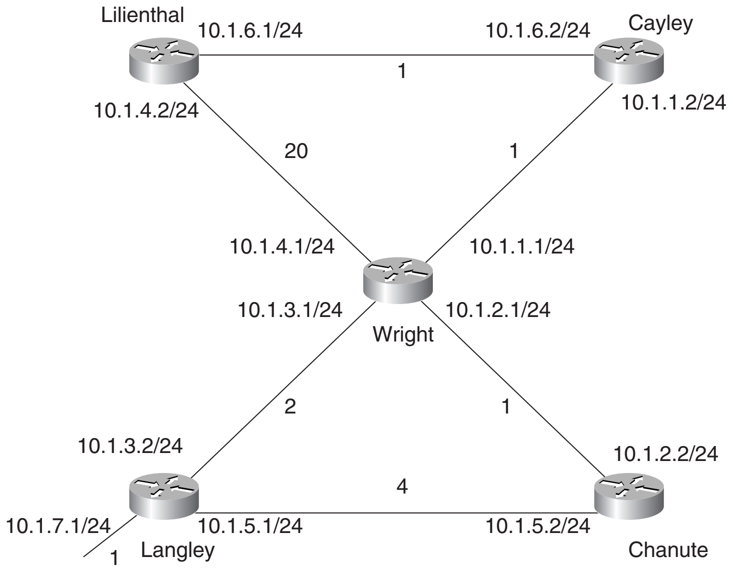
It should be pointed out that although the delay parameters used here sacrifice realism for simplicity, the way the metrics are manipulated is realistic. Many parameters are calculated from an interface's **bandwidth** specification; some, such as the **ip bandwidth-percent eigrp**, apply directly to EIGRP. Others, such as OSPF cost, do not. As a result, changes of the configured bandwidth should be avoided except to set serial links to their actual bandwidth. If interface metrics need to be manipulated to influence EIGRP (or IGRP) routing, use **delay**. Many unexpected headaches can be avoided.

In Example 7-7, the command **show ip eigrp topology** is used to observe the topology table of router Langley. Each of the seven subnets shown in Figure 7-5 is listed, along with the feasible successors for the subnets. For example, the feasible successors for subnet 10.1.6.0 are 10.1.3.1 (Wright) and 10.1.5.2 (Chanute), via interfaces S0 and S1, respectively.

[10] Actually, the interface is not explicitly recorded in the route table. Rather, it is an attribute of the neighbor itself. This convention implies that the same router, seen across multiple parallel links, will be viewed by EIGRP as multiple neighbors.

[11] Several of the illustrations in this and the following section, and in the network example used throughout, are adapted from Dr. Garcia-Luna-Aceves's "Loop-Free Routing Using Diffusing Computations," with his permission.

**Figure 7-5** *The examples and illustrations of this and the next two subsections are based on this EIGRP network.*



**Example 7-7** *Topology table of router Langley.*

```
Langley#show ip eigrp topology
IP-EIGRP Topology Table for process 1

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status

P 10.1.3.0/24, 1 successors, FD is 512
         via Connected, Serial0
P 10.1.2.0/24, 1 successors, FD is 768
         via 10.1.3.1 (768/256), Serial0
         via 10.1.5.2 (1280/256), Serial1
P 10.1.1.0/24, 1 successors, FD is 768
         via 10.1.3.1 (768/256), Serial0
         via 10.1.5.2 (1536/512), Serial1
P 10.1.7.0/24, 1 successors, FD is 256
         via Connected, Ethernet0
P 10.1.6.0/24, 1 successors, FD is 1024
         via 10.1.3.1 (1024/512), Serial0
         via 10.1.5.2 (1792/768), Serial1
P 10.1.5.0/24, 1 successors, FD is 1024
         via Connected, Serial1
P 10.1.4.0/24, 1 successors, FD is 5632
         via 10.1.3.1 (5632/5120), Serial0
         via 10.1.5.2 (6400/5376), Serial1
Langley#
```

Two metrics in parentheses are also associated with each feasible successor. The first number is the locally calculated metric from Langley to the destination. The second number is the metric advertised by the neighbor. For example, in Figure 7-5 the metric from Langley to subnet 10.1.6.0 via Wright is 256 x (2 + 1 + 1) = 1024, and the metric advertised by Wright for that destination is 256 x (1 + 1) = 512. The two metrics for the same destination via Chanute are 256 x (4 + 1 + 1 + 1) = 1792 and 256 x (1 + 1 + 1) = 768.

The lowest metric from Langley to subnet 10.1.6.0 is 1024, so that is the FD. Example 7-8 shows Langley's route table, with the chosen successors.

**Example 7-8** *Langley's route table shows that a single successor has been chosen for each known destination, based on the lowest metric distance.*

```
Langley#show ip route
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
       U - per-user static route
Gateway of last resort is not set
     10.0.0.0/8 is subnetted, 7 subnets
C       10.1.3.0 is directly connected, Serial0
D       10.1.2.0 [90/768] via 10.1.3.1, 00:32:06, Serial0
D       10.1.1.0 [90/768] via 10.1.3.1, 00:32:07, Serial0
C       10.1.7.0 is directly connected, Ethernet0
D       10.1.6.0 [90/1024] via 10.1.3.1, 00:32:07, Serial0
C       10.1.5.0 is directly connected, Serial1
D       10.1.4.0 [90/5632] via 10.1.3.1, 00:32:07, Serial0
Langley#
```

Langley has only one successor for every route. The topology table of Cayley (Example 7-9) shows that there are two successors for 10.1.4.0 because the locally calculated metric for both routes matches the FD. Both routes are entered into the route table (Example 7-10), and Cayley will perform equal-cost load balancing.

**Example 7-9** *Topology table of Cayley, showing two successors to subnet 10.1.4.0.*

```
Cayley#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
P 10.1.3.0/24, 1 successors, FD is 768
         via 10.1.1.1 (768/512), Ethernet0
P 10.1.2.0/24, 1 successors, FD is 512
         via 10.1.1.1 (512/256), Ethernet0
P 10.1.1.0/24, 1 successors, FD is 256
         via Connected, Ethernet0
P 10.1.7.0/24, 1 successors, FD is 1024
         via 10.1.1.1 (1024/768), Ethernet0
P 10.1.6.0/24, 1 successors, FD is 256
         via Connected, Serial0
P 10.1.5.0/24, 1 successors, FD is 1536
         via 10.1.1.1 (1536/1280), Ethernet0
```

**Example 7-9** *Topology table of Cayley, showing two successors to subnet 10.1.4.0. (Continued)*

```
P 10.1.4.0/24, 2 successors, FD is 5376
         via 10.1.6.1 (5376/5120), Serial0
         via 10.1.1.1 (5376/5120), Ethernet0
Cayley#
```

**Example 7-10** *Equal-cost load sharing will be performed between the two successors to 10.1.4.*

```
Cayley#show ip route
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
Gateway of last resort is not set
     10.0.0.0/24 is subnetted, 7 subnets
D       10.1.3.0 [90/768] via 10.1.1.1, 00:01:19, Ethernet0
D       10.1.2.0 [90/512] via 10.1.1.1, 00:01:19, Ethernet0
C       10.1.1.0 is directly connected, Ethernet0
D       10.1.7.0 [90/1024] via 10.1.1.1, 00:01:19, Ethernet0
C       10.1.6.0 is directly connected, Serial0
D       10.1.5.0 [90/1536] via 10.1.1.1, 00:01:19, Ethernet0
D       10.1.4.0 [90/5376] via 10.1.1.1, 00:01:19, Ethernet0
                 [90/5376] via 10.1.6.1, 00:01:19, Serial0
Cayley#
```

The topology table of Chanute (Example 7-11) shows several routes for which there is only one feasible successor. For example, the route to 10.1.6.0 has an FD of 768, and Wright (10.1.2.1) is the only feasible successor. Langley has a route to 10.1.6.0, but its metric is 256 x (2 + 1 + 1) = 1024, which is greater than the FD. Therefore, Langley's route to 10.1.6.0 does not satisfy the FC, and Langley does not qualify as a feasible successor.

**Example 7-11** *Several of the subnets reachable from Chanute have only one feasible successor.*

```
Chanute#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
P 10.1.3.0/24, 1 successors, FD is 768
         via 10.1.2.1 (768/512), Ethernet0
         via 10.1.5.1 (1536/512), Serial0
P 10.1.2.0/24, 1 successors, FD is 256
         via Connected, Ethernet0
P 10.1.1.0/24, 1 successors, FD is 512
         via 10.1.2.1 (512/256), Ethernet0
P 10.1.7.0/24, 1 successors, FD is 1024
         via 10.1.2.1 (1024/768), Ethernet0
         via 10.1.5.1 (1280/256), Serial0
P 10.1.6.0/24, 1 successors, FD is 768
         via 10.1.2.1 (768/512), Ethernet0
P 10.1.5.0/24, 1 successors, FD is 1024
         via Connected, Serial0
```

*continues*

**Example 7-11** *Several of the subnets reachable from Chanute have only one feasible successor. (Continued)*

```
P 10.1.4.0/24, 1 successors, FD is 5376
         via 10.1.2.1 (5376/5120), Ethernet0
Chanute#
```

If a feasible successor advertises a route for which the locally calculated metric is lower than the metric via the present successor, the feasible successor will become the successor. The following conditions can cause this situation to occur:

- A newly discovered route
- The cost of a successor's route increasing beyond that of a feasible successor
- The cost of a feasible successor's route decreasing to below the cost of the successor's route

For example, Example 7-12 shows that Lilienthal's successor to subnet 10.1.3.0 is Cayley (10.1.6.2). Suppose the cost of the link between Lilienthal and Wright is decreased to one. Wright (10.1.4.1) is advertising a distance of 512 to subnet 10.1.3.0; with the new cost of the link to Wright, Lilienthal's locally calculated metric to the subnet via that router is now 768. Wright will replace Cayley as the successor to subnet 10.1.3.0.

**Example 7-12** *Topology table for Lilienthal.*

```
Lilienthal#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - Reply status
P 10.1.3.0/24, 1 successors, FD is 1024
         via 10.1.6.2 (1024/768), Serial0
         via 10.1.4.1 (5632/512), Serial1
P 10.1.2.0/24, 1 successors, FD is 768
         via 10.1.6.2 (768/512), Serial0
         via 10.1.4.1 (5376/256), Serial1
P 10.1.1.0/24, 1 successors, FD is 512
         via 10.1.6.2 (512/256), Serial0
         via 10.1.4.1 (5376/256), Serial1
P 10.1.7.0/24, 1 successors, FD is 1280
         via 10.1.6.2 (1280/1024), Serial0
         via 10.1.4.1 (5888/768), Serial1
P 10.1.6.0/24, 1 successors, FD is 256
         via Connected, Serial0
P 10.1.5.0/24, 1 successors, FD is 1792
         via 10.1.6.2 (1792/1536), Serial0
         via 10.1.4.1 (6400/1280), Serial1
P 10.1.4.0/24, 1 successors, FD is 5120
         via Connected, Serial1
Lilienthal#
```

Next, suppose Lilienthal discovers a new neighbor that is advertising a distance of 256 to subnet 10.1.3.0. This distance is lower than the FD, so the new neighbor will become a feasible successor. Suppose further that the cost of the link to the new neighbor is 256. Lilienthal's locally calculated metric to 10.1.3.0 via the new neighbor will be 512. This metric is lower than the distance via Wright, so the new neighbor will become the successor to 10.1.3.0.

Feasible successors are important because they reduce the number of diffusing computations and therefore increase performance. Feasible successors also contribute to lower reconvergence times. If a link to a successor fails, or if the cost of the link increases beyond the FD, the router will first look into its topology table for a feasible successor. If one is found, it will become the successor; this selection normally occurs in the sub-second range. The router will begin a diffusing computation only if a feasible successor cannot be found. The key to successful EIGRP design, then, is ensuring that a feasible successor always exists for all destinations.

The following section gives a more formal set of rules for when and how a router will search for feasible successors. This set of rules is called the *DUAL finite state machine*.

## DUAL Finite State Machine

When an EIGRP router is performing no diffusing computations, each route is in the *passive state*. Referring to any of the topology tables in the previous section, a key to the left of each route indicates a passive state.

A router will reassess its list of feasible successors for a route, as described in the last section, any time an *input event* occurs. An input event can be the following:

- A change in the cost of a directly connected link
- A change in the state (up or down) of a directly connected link
- The reception of an update packet
- The reception of a query packet
- The reception of a reply packet

The first step in its reassessment is a *local computation* in which the distance to the destination is recalculated for all feasible successors. The possible results are

- If the feasible successor with the lowest distance is different from the existing successor, the feasible successor will become the successor.
- If the new distance is lower than the FD, the FD will be updated.
- If the new distance is different from the existing distance, updates will be sent to all neighbors.

While the router is performing a local computation, the route remains in the passive state. If a feasible successor is found, an update is sent to all neighbors and no state change occurs.

If a feasible successor cannot be found in the topology table, the router will begin a diffusing computation and the route will change to the *active state*. Until the diffusing computation is completed and the route transitions back to the passive state, the router cannot

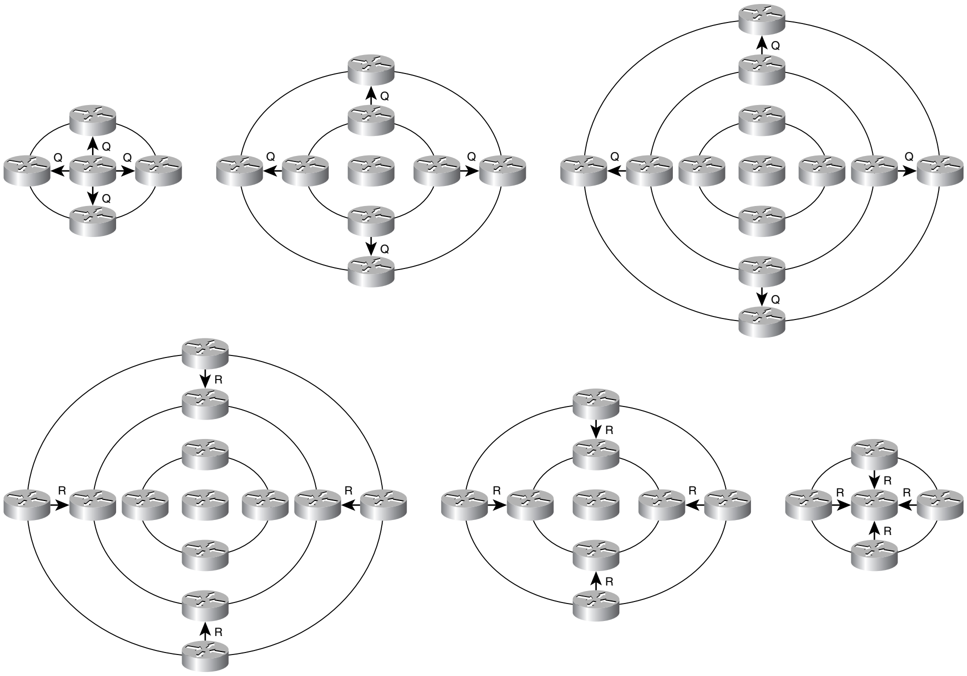
- Change the route's successor
- Change the distance it is advertising for the route

- Change the route's FD
- Begin another diffusing computation for the route

A router begins a diffusing computation by sending queries to all of its neighbors (Figure 7-6). The query will contain the new locally calculated distance to the destination. Each neighbor, upon receipt of the query, will perform its own local computation:

- If the neighbor has one or more feasible successors for the destination, it will send a reply to the originating router. The reply will contain that neighbor's minimum locally calculated distance to the destination.
- If the neighbor does not have a feasible successor, it too will change the route to the active state and will begin a diffusing computation.

**Figure 7-6** *A diffusing computation grows as queries are sent and shrinks as replies are received.*



For each neighbor to which a query is sent, the router will set a *reply status flag* (r) to keep track of all outstanding queries. The diffusing computation is complete when the router has received a reply to every query sent to every neighbor.

In some cases, a router does not receive a reply to every query sent. For example, this might happen in large networks with many low-bandwidth or low-quality links. At the beginning of the diffusing computation, an Active timer is set for three minutes. If all expected replies are not received before the Active time expires, the route is declared *stuck-in-active* (SIA). The neighbor or neighbors that did not reply will be removed from the neighbor table, and the diffusing computation will consider the neighbor to have responded with an infinite metric.

The default three-minute Active time can be changed or disabled with the command **timers active-time**. The deletion of a neighbor because of a lost query obviously can have disruptive results, and SIAs should never occur in a stable, well-designed network. The troubleshooting section of this chapter discusses SIAs in more detail. That same section describes a recent enhancement to the SIA procedures, using two new EIGRP messages—SIA Query and SIA Reply—that both reduce the chance of occurrences of SIAs and, when they do occur, make them easier to troubleshoot.

At the completion of the diffusing computation, the originating router will set FD to infinity to ensure that any neighbor replying with a finite distance to the destination will meet the FC and become a feasible successor. For each of these replies, a metric is calculated based on the distance advertised in the reply plus the cost of the link to the neighbor who sent the reply. A successor is selected based on the lowest metric, and FD is set to this metric. Any feasible successors that do not satisfy the FC for this new FD will be removed from the topology table. Note that a successor is not chosen until all replies have been received.

Because there are multiple types of input events that can cause a route to change state, some of which might occur while a route is active, DUAL defines multiple active states. A *query origin flag* (O) is used to indicate the current state. Figure 7-7 and Table 7-2 show the complete DUAL finite state machine.

**Figure 7-7** *The DUAL finite state machine. The query origin flag (O) marks the current state of the diffusing calculation. See Table 7-2 for a description of each input event (IE).*

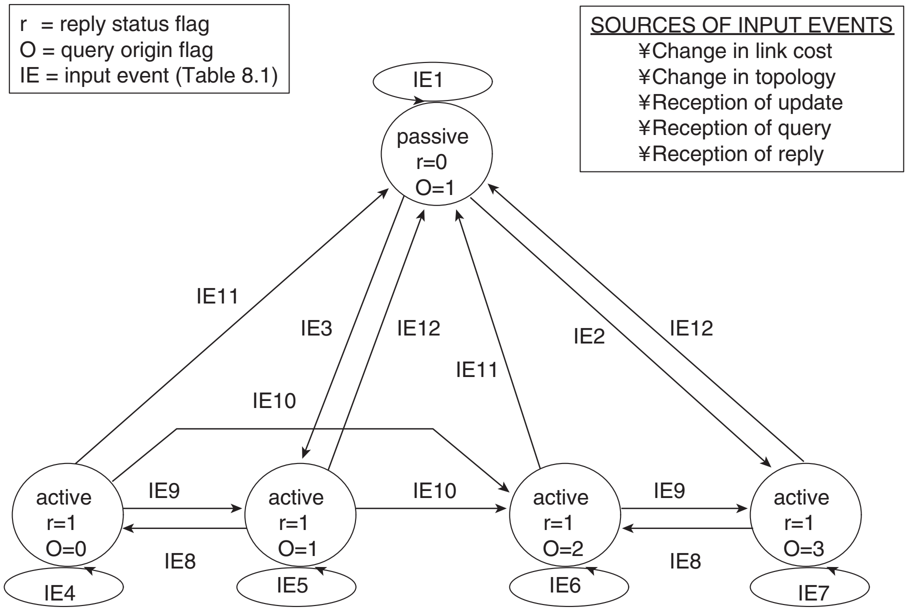


**Table 7-2** *Input events for the DUAL finite state machine.*

| Input Event | Description |
|---|---|
| IE1 | Any input event for which FC is satisfied or the destination is unreachable |
| IE2 | Query received from the successor; FC not satisfied |
| IE3 | Input event other than a query from the successor; FC not satisfied |
| IE4 | Input event other than last reply or a query from the successor |
| IE5 | Input event other than last reply, a query from the successor, or an increase in distance to destination |
| IE6 | Input event other than last reply |
| IE7 | Input event other than last reply or increase in distance to destination |
| IE8 | Increase in distance to destination |
| IE9 | Last reply received; FC not met with current FD |
| IE10 | Query received from the successor |
| IE11 | Last reply received; FC met with current FD |
| IE12 | Last reply received; set FD to infinity |

Two examples can help clarify the DUAL process. Figure 7-8 shows the example network, focusing only on each router's path to subnet 10.1.7.0; refer to Figure 7-5 for specific addresses. On the data links, an arrow indicates the successor each router is using to reach 10.1.7.0. In parentheses are each router's locally calculated distance to the subnet, the router's FD, the reply status flag (r), and the query origin flag (O), respectively. Active routers are indicated with a circle in subsequent figures and examples using this network.

## Diffusing Computation: Example 1

This example focuses only on Cayley and its route to subnet 10.1.7.0. In Figure 7-9, the link between Cayley and Wright (10.1.1.1) has failed. EIGRP interprets the failure as a link with an infinite distance.[12] Cayley checks its topology table for a feasible successor to 10.1.7.0 and finds none (refer to Example 7-9).

Cayley's route becomes active (Figure 7-10). The distance and the FD of the route are changed to unreachable, and a query containing the new distance is sent to Cayley's neighbor, Lilienthal. Cayley's reply status flag for Lilienthal is set to one, indicating that a reply is expected. Because the input event was not the reception of a query (IE3), O=1.

[12] An infinite distance is indicated by a delay of 0xFFFFFFFF, or 4294967295.

**Figure 7-8** *All routes to subnet 10.1.7.0 are in the passive state, indicated by r = 0 and O = 1.*

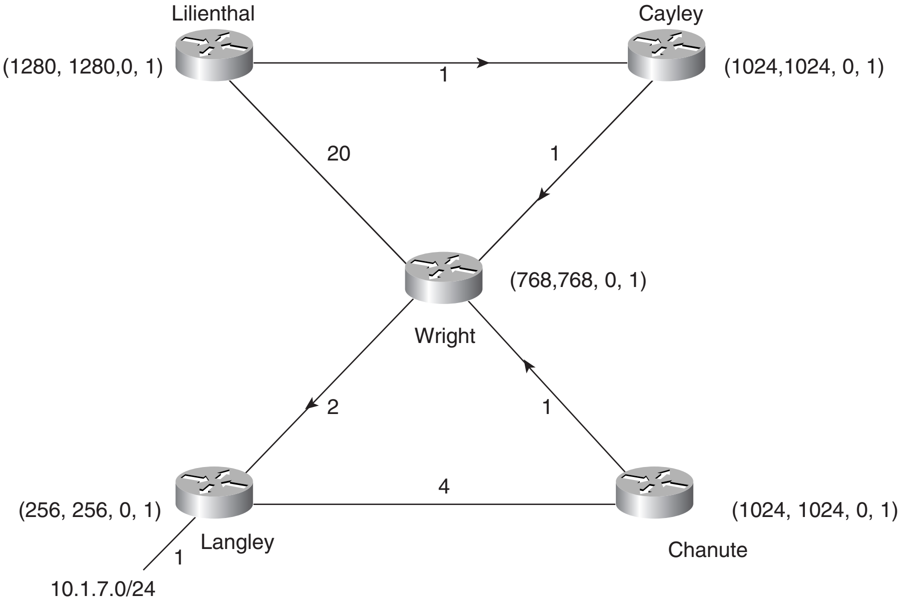


**Figure 7-9** *The link between Wright and Cayley has failed, and Cayley does not have a feasible successor to subnet 10.1.7.0.*

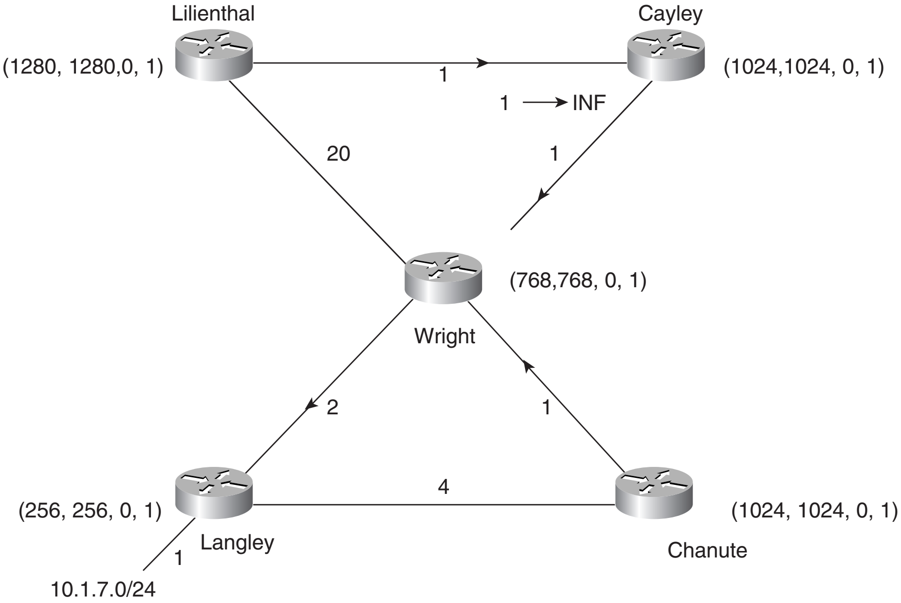
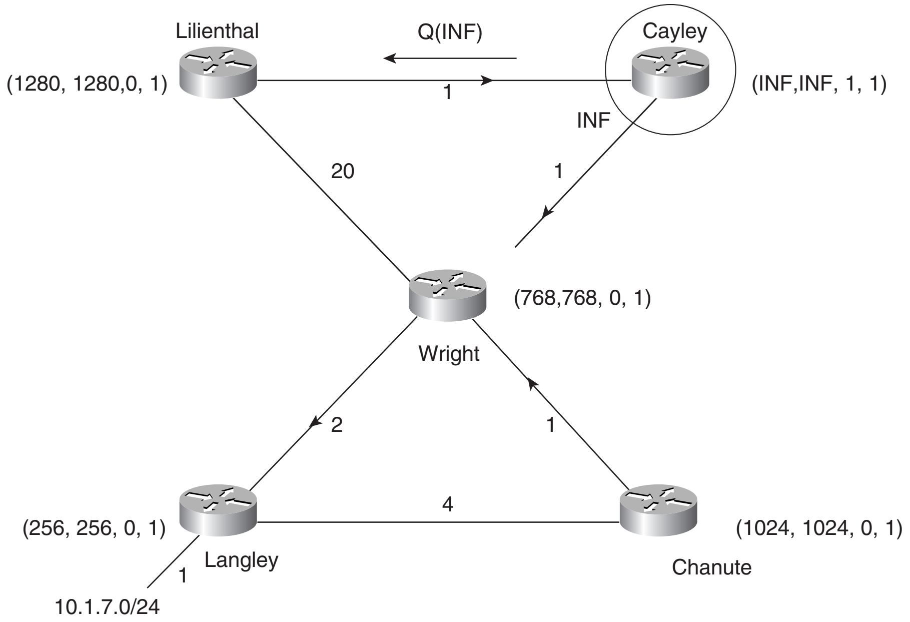


**Figure 7-10** *Cayley's route to 10.1.7.0 transitions to active, and Lilienthal is queried for a feasible successor.*



Upon receipt of the query, Lilienthal performs a local computation (Figure 7-11). Because Lilienthal has a feasible successor for 10.1.7.0 (refer to Example 7-12), the route does not become active. Wright becomes the new successor, and a reply is sent with Lilienthal's distance to 10.1.7.0 via Wright. Because the distance to 10.1.7.0 has increased and the route did not become active, the FD is unchanged at Lilienthal.

Upon receipt of the reply from Lilienthal, Cayley sets r=0 and the route becomes passive (Figure 7-12). Lilienthal becomes the new successor, and the FD is set to the new distance. Finally, an update is sent to Lilienthal with Cayley's locally calculated metric. Lilienthal will also send an update advertising its new metric.

EIGRP packet activity can be observed with the debug command **debug eigrp packets**. By default, all EIGRP packets are displayed. Because Hellos and ACKs can make the debug output hard to follow, the command allows the use of optional keywords so that only specified packet types are displayed. In Example 7-13, **debug eigrp packets query reply update** is used to observe the packet activity at Cayley for the events described in this example.

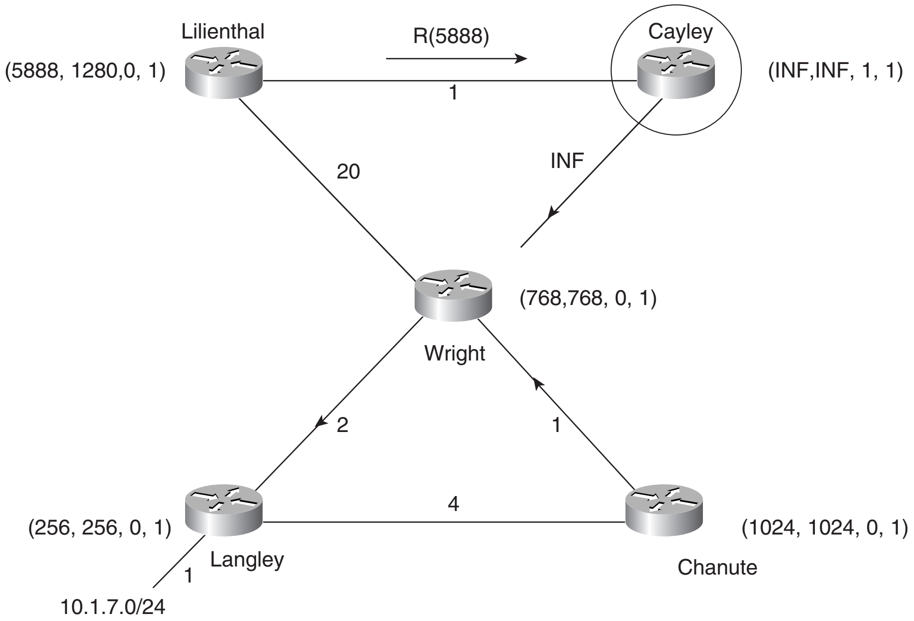


**Figure 7-11** *Lilienthal has a feasible successor to 10.1.7.0. A local computation is performed, a reply is sent to Cayley with the distance via Wright, and an update is sent to Wright.*

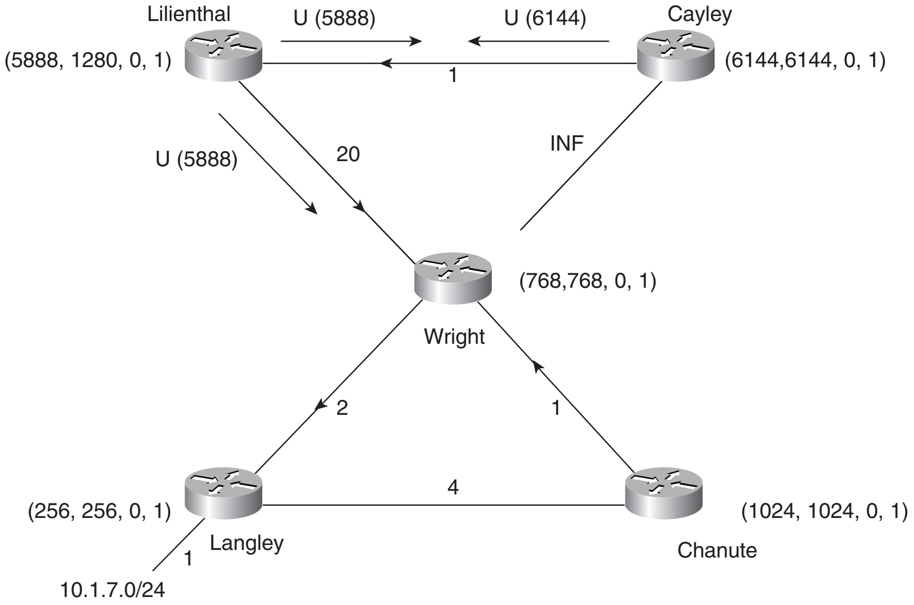


**Figure 7-12** *Cayley's route to 10.1.7.0 becomes passive, and an update is sent to Lilienthal.*

**Example 7-13** *The EIGRP packet events described in this example can be observed in these debug messages.*

```
Cayley#debug eigrp packet update query reply
EIGRP Packets debugging is on
    (UPDATE, QUERY, REPLY)
B#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to down
EIGRP: Enqueueing QUERY on Serial0 iidbQ un/rely 0/1 serno 45-49
EIGRP: Enqueueing QUERY on Serial0 nbr 10.1.6.1 iidbQ un/rely 0/0 peerQ un/rely
0/0 serno 45-49
EIGRP: Sending QUERY on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 45/64 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
45-49
EIGRP: Received REPLY on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 65/45 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Enqueueing UPDATE on Serial0 iidbQ un/rely 0/1 serno 50-54
EIGRP: Enqueueing UPDATE on Serial0 nbr 10.1.6.1 iidbQ un/rely 0/0 peerQ un/rely
 0/0 serno 50-54
EIGRP: Sending UPDATE on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 46/66 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
50-54
EIGRP: Received UPDATE on Serial0 nbr 10.1.6.1
  AS 1, Flags 0x0, Seq 67/46 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
```

*Flags*, in the debug messages, indicate the state of the flags in the EIGRP packet header (see the section "EIGRP Packet Header" later in this chapter). 0x0 indicates that no flags are set. 0x1 indicates that the *initialization* bit is set. This flag is set when the enclosed route entries are the first in a new neighbor relationship. 0x2 indicates that the *conditional receive* bit is set. This flag is used in the proprietary Reliable Multicasting algorithm:

- **Seq** is the Packet Sequence Number/Acknowledged Sequence Number.
- **idbq** indicates packets in the input queue/packets in the output queue of the interface.
- **iidbq** indicates unreliable multicast packets awaiting transmission/reliable multicast packets awaiting transmission on the interface.
- **peerQ** indicates unreliable unicast packets awaiting transmission/reliable unicast packets awaiting transmission on the interface.
- **serno** is a pointer to a doubly linked serial number for the route. This is used by an internal (and proprietary) mechanism for tracking the correct route information in a rapidly changing topology.

## Diffusing Computation: Example 2

This example focuses on Wright and its route to subnet 10.1.7.0. Although the combination of input events portrayed here (the delay of a link changing twice during a diffusing computation) is unlikely to occur in real life, the example shows how DUAL handles multiple metric changes.

In Figure 7-13, the cost of the link between Wright and Langley changes from 2 to 10. The distance to 10.1.7.0 via Langley now exceeds Wright's FD, causing that router to begin a local computation. The metric is updated, and Wright sends updates to all its neighbors except the neighbor on the link whose cost changed (Figure 7-14).

**Figure 7-13** *Cayley's route to 10.1.7.0 becomes passive, and an update is sent to Lilienthal.*

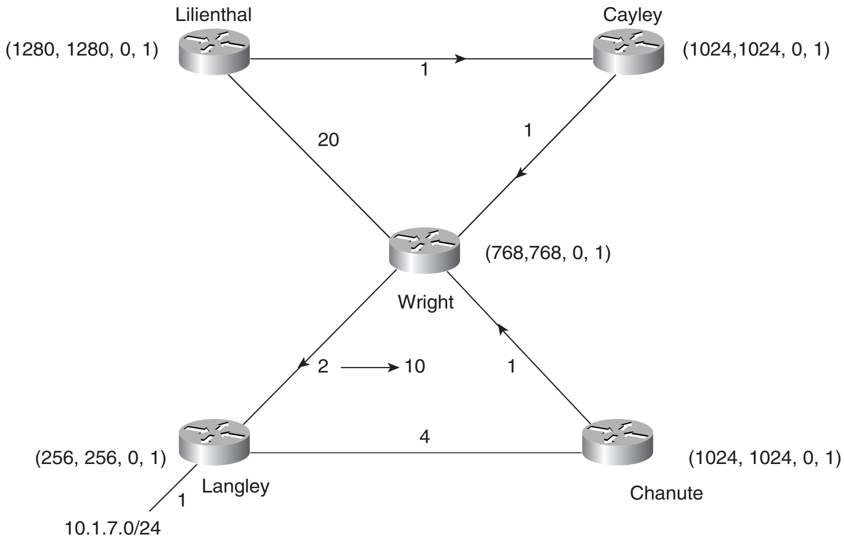
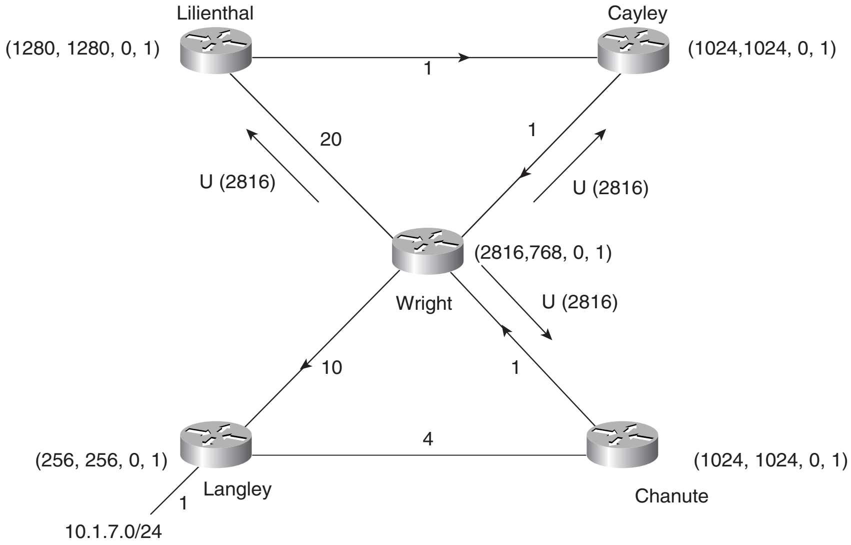
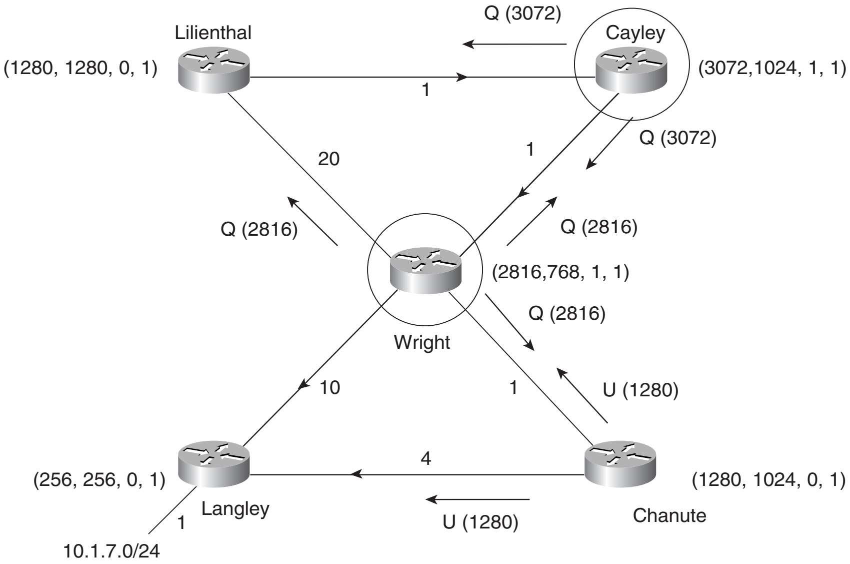


**Figure 7-14** *Wright sends updates containing the new metric to all neighbors except Langley.*



Note that Langley was the only feasible successor to subnet 10.1.7.0 because Chanute's locally calculated metric is higher than Wright's FD (1024 > 768). The metric increase on the Wright-Langley link causes Wright to look in its topology table for a new successor. Because the only feasible successor that Wright can find in its topology table is Langley, the route becomes active. Queries are sent to the neighbors (Figure 7-15).

**Figure 7-15** *Wright's route to 10.1.7.0 becomes active, and it queries its neighbors for a feasible successor. In response to the earlier update from Wright, Cayley makes its route active and queries its neighbors; also, Chanute changes its metric and sends updates.*



At the same time, the updates sent by Wright in Figure 7-14 cause Cayley, Lilienthal, and Chanute to perform a local calculation.

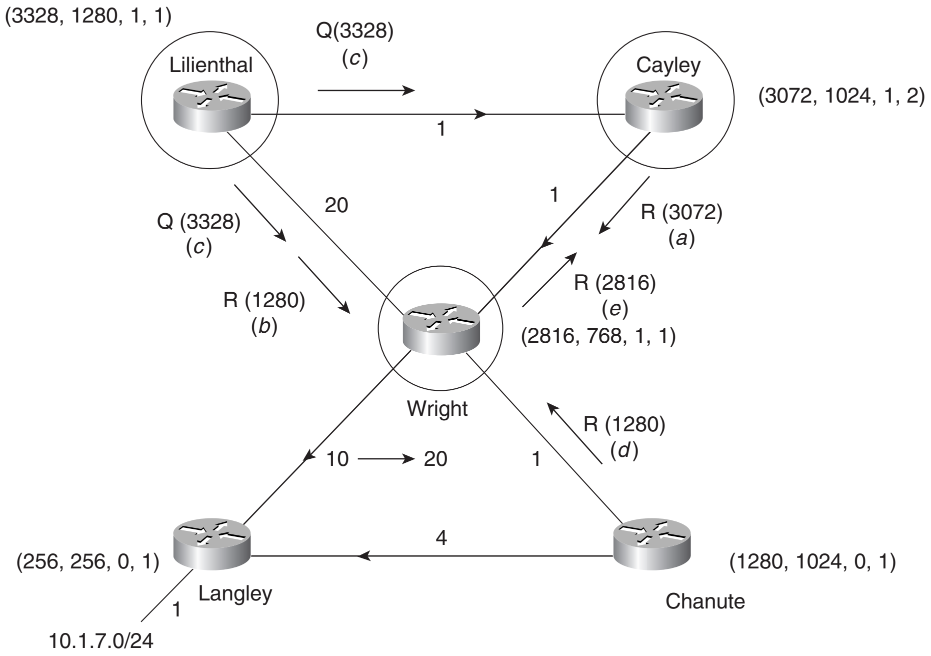
At Cayley, the route via Wright now exceeds Cayley's FD (2816 > 1024). The route goes active and queries are sent to the neighbors.

Lilienthal is using Cayley as a successor and in Figure 7-15 has not yet received the query from Cayley. Therefore, Lilienthal merely recalculates the metric of the path via Wright, finds that it no longer meets the FC, and drops the path from the topology table.

At Chanute, Wright is the successor. Because Wright's advertised distance no longer meets the FC at Chanute (2816 > 1024) and because Chanute does have a feasible successor (refer to Example 7-11), Wright is deleted from Chanute's topology table. Langley becomes the successor at Chanute; the metric is updated, and Chanute sends updates to its neighbors (refer to Figure 7-15). The route at Chanute never becomes active.

Cayley, Lilienthal, and Chanute each respond differently to the queries from Wright (Figure 7-16).

**Figure 7-16** *Cayley (a) replies to Wright's query. Lilienthal (b) replies to Wright's query and (c) goes active for the route, sending queries in response to Cayley's query. Chanute (d) replies to Wright's query. Wright (e) replies to Cayley's query.*



Cayley is already active. Because the input event is a query from the successor, the query origin flag will be 2 (O=2) (refer to Figure 7-7 and Table 7-2).

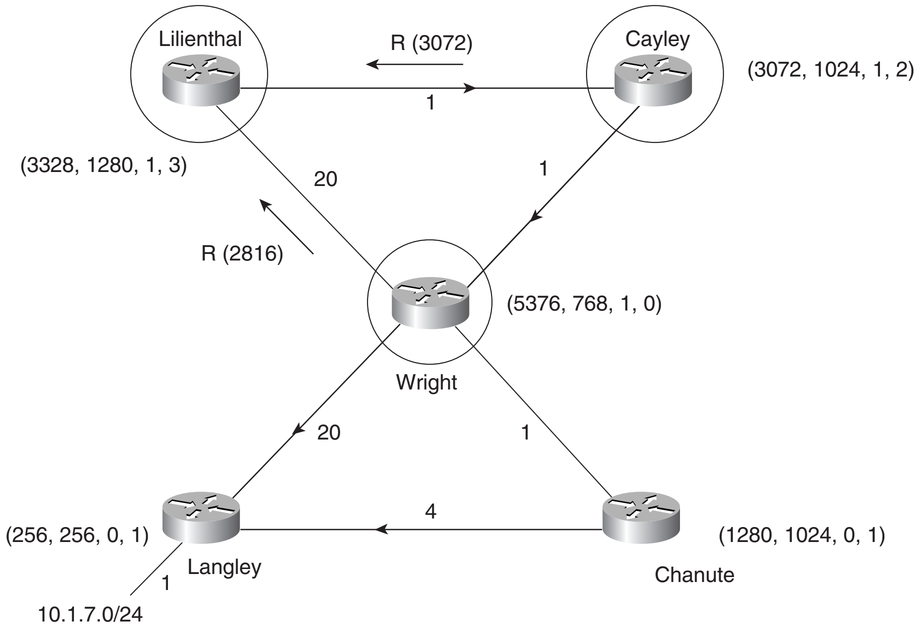
Lilienthal, upon the receipt of Wright's query, sends a response with its distance via Cayley. However, just after the reply is sent, Lilienthal receives the query from Cayley. The FD is exceeded, the metric is updated, and the route goes active. Lilienthal queries its neighbors.

Chanute, which has already switched to Langley as its successor, merely sends a reply.

While all this is going on, Figure 7-16 shows that the cost of the link between Wright and Langley again increases, from 10 to 20. Wright will recalculate the metric to 10.1.7.0 based on this new cost, but because the route is active, neither the FD nor the distance it advertises will change until the route becomes passive.

According to Figure 7-7 and Table 7-2, an increase in the distance to the destination while the route is active will cause O=0 (Figure 7-17). Wright responds to the query from Lilienthal. The distance it reports is the distance it had when the route first became active (remember, the advertised distance cannot change while the route is active). Cayley also sends a reply to Lilienthal's query.

**Figure 7-17** *Wright cannot change the metric it advertises until the route becomes passive.*



Lilienthal, having received replies to all its queries, will transition the route to passive (Figure 7-18). A new FD is set for the route. Cayley remains the successor because its advertised route is lower than the FD at Lilienthal. Lilienthal also sends a reply to Cayley's query.

Figure 7-18 also shows that the distance has changed again, from 20 to 15. Wright recalculates its local distance for the route again, to 4096 (Figure 7-19). If it were to receive a query before going passive, the route would still be advertised with a distance of 2816—the distance when the route went active.

When Cayley receives the reply to its query, its route to 10.1.7.0 also becomes passive (Figure 7-19) and a new FD is set. Although Wright's locally calculated metric is 4096, the last metric it advertised was 2816. Therefore, Wright meets the FC at Cayley and becomes the successor to 10.1.7.0. A reply is sent to Wright.

In Figure 7-20, Wright has received a reply to every query it sent, and its route becomes passive. It chooses Chanute as its new successor and changes the FD to the sum of Chanute's advertised distance and the cost of the link to that neighbor. Wright sends an update to all its neighbors, advertising the new locally calculated metric.

Cayley is already using Wright as the successor. When it receives the update from Wright with a lower cost, it changes its locally calculated metric and FD accordingly and updates its neighbors (Figure 7-21).

The update from Cayley has no effect at Wright because it does not satisfy the FC there. At Lilienthal the update causes a local computation. Lilienthal lowers the metric, lowers the FD, and updates its neighbors (Figure 7-22).

**Figure 7-18** *Having received the last expected reply, Lilienthal changes its route to the passive state (r=0, O=1).*

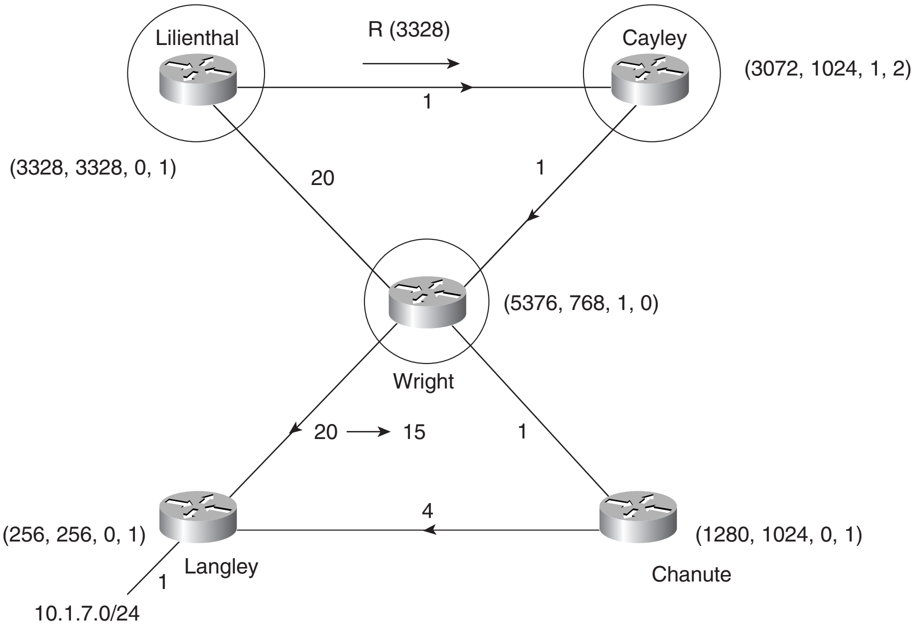


**Figure 7-19** *Having received its last expected reply, Cayley changes its route to the passive state.*

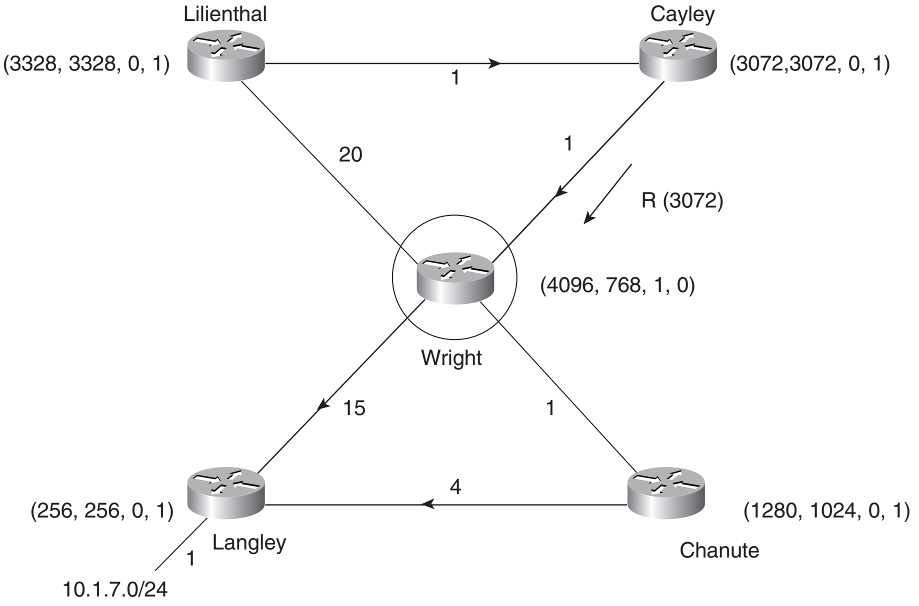


**Figure 7-20** *Wright transitions to passive, chooses Chanute as the successor, changes the FD, and updates all neighbors.*

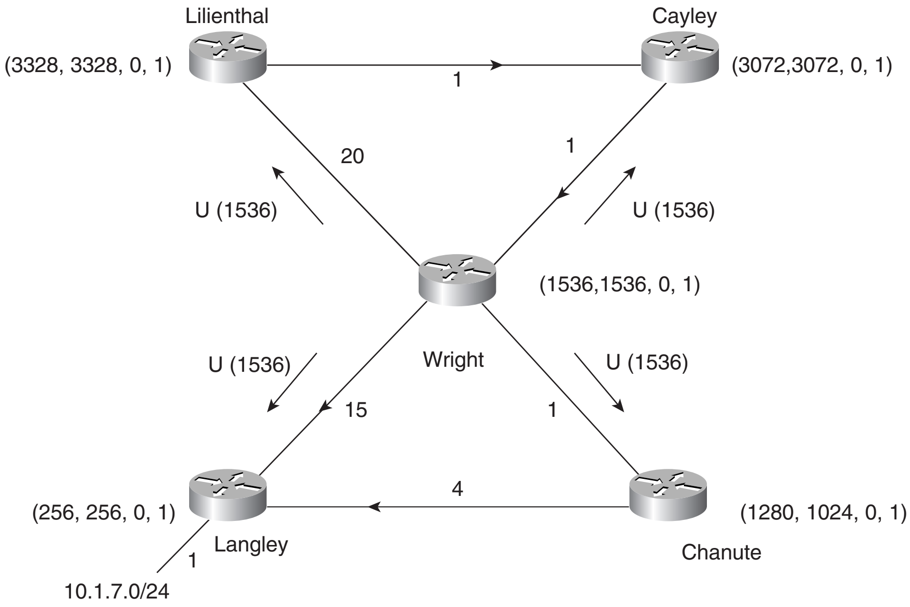


**Figure 7-21** *Cayley recalculates its metric, changes the FD based on the lower cost advertised by Wright, and updates its neighbors.*

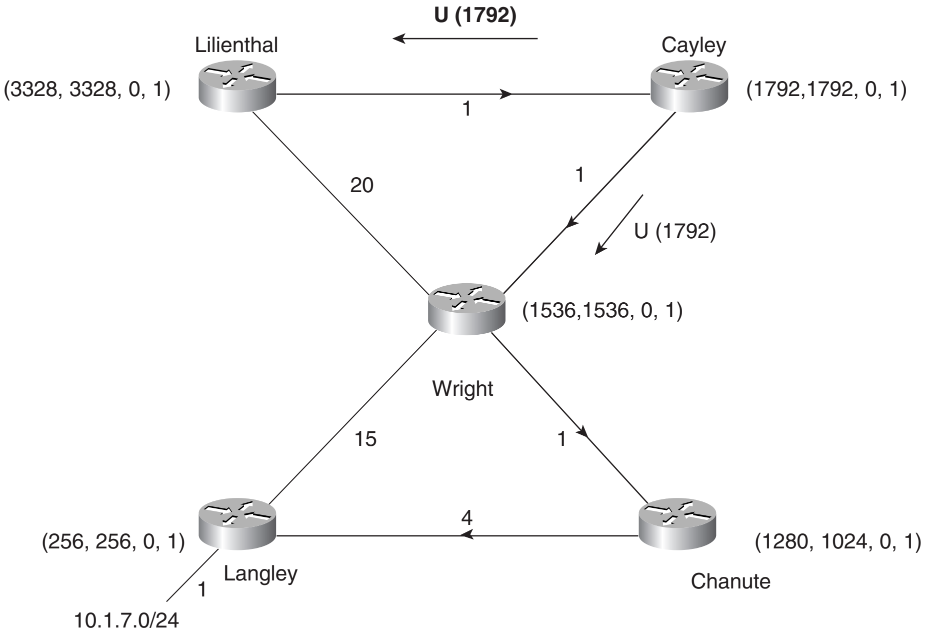
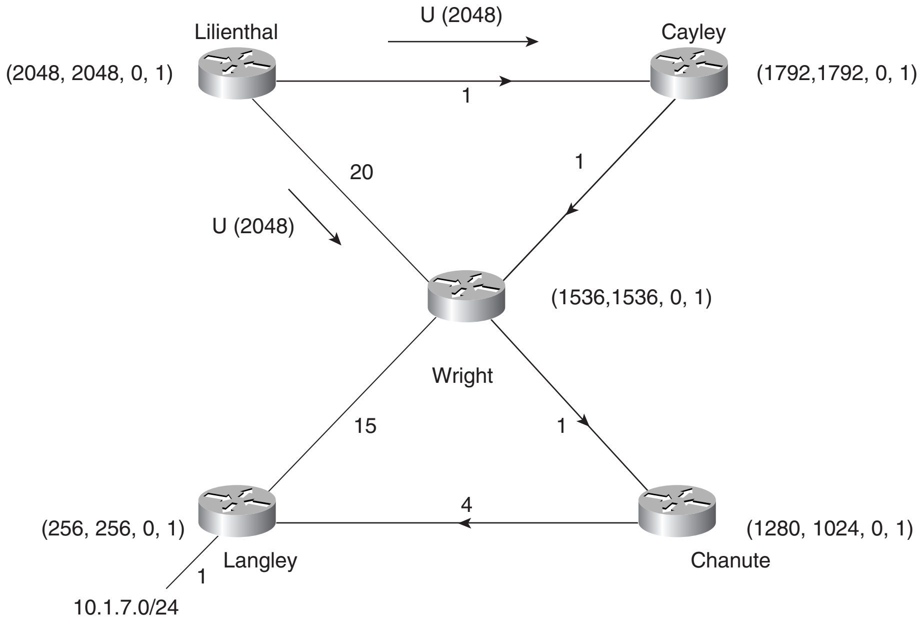


**Figure 7-22** *Lilienthal recalculates its metric, changes the FD based on the update from Cayley, and updates its neighbors.*



Although they are rather elaborate and might take several readings to fully understand, this and the previous example contain the important core behavior of diffusing computations:

- Any time an input event occurs, perform a local calculation.
- If one or more feasible successors are found in the topology table, take the ones with the lowest metric cost as the successors.
- If no feasible successor can be found, make the route active and query the neighbors for a feasible successor.
- Keep the route active until all queries are answered by reply or by the expiration of the active timer.
- If the diffusing calculation does not result in the discovery of a feasible successor, declare the destination unreachable.

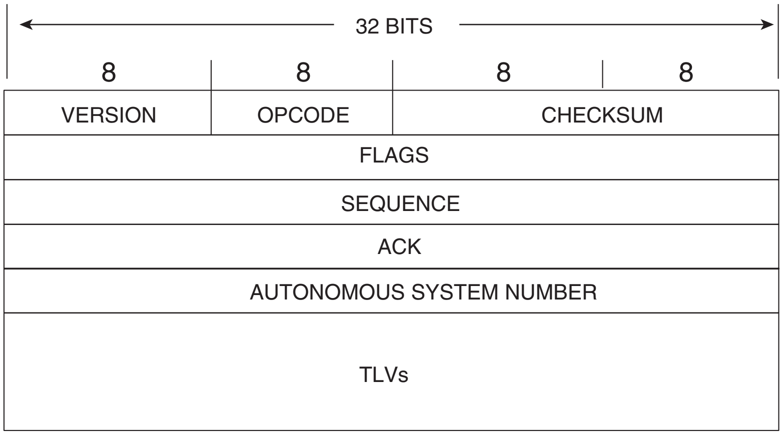
## EIGRP Packet Formats

The IP header of an EIGRP packet specifies protocol number 88, and the maximum length of the packet will be the IP maximum transmission unit (MTU) of the interface on which it is transmitted—usually 1500 octets. Following the IP header is an EIGRP header followed by various *Type/Length/Value* (TLV) triplets. These TLVs will not only carry the route entries but also may provide fields for the management of the DUAL process, multicast sequencing, and IOS software versions.

## EIGRP Packet Header

Figure 7-23 shows the EIGRP header, which begins every EIGRP packet.

**Figure 7-23** *EIGRP packet header.*



- **Version** specifies the particular version of the originating EIGRP process. The version of the EIGRP process itself has not changed since its release.
- **Opcode** specifies the EIGRP packet type, as shown in Table 7-3. Although the IPX SAP packet type is included in the table, a discussion of IPX EIGRP is outside the scope of this book.

**Table 7-3** *EIGRP packet types.*

| Opcode | Type |
|---|---|
| 1 | Update |
| 3 | Query |
| 4 | Reply |
| 5 | Hello |
| 6 | IPX SAP |
| 10 | SIA Query |
| 11 | SIA Reply |

- **Checksum** is a standard IP checksum. It is calculated for the entire EIGRP packet, excluding the IP header.
- **Flags** currently include just two flags. The right-most bit is *Init*, which when set (0x00000001) indicates that the enclosed route entries are the first in a new neighbor relationship. The second bit (0x00000002) is the Conditional Receive bit, used in the proprietary Reliable Multicasting algorithm.

- **Sequence** is the 32-bit sequence number used by the RTP.
- **ACK** is the 32-bit sequence number last heard from the neighbor to which the packet is being sent. A Hello packet with a nonzero ACK field will be treated as an ACK packet rather than as a Hello. Note that an ACK field will only be nonzero if the packet itself is unicast because acknowledgments are never multicast.
- **Autonomous System Number** is the identification number of the EIGRP domain.

Following the header are the TLVs, whose various types are listed in Table 7-4. IPX and AppleTalk types are included, although they are not discussed in this book. Each TLV includes one of the two-octet type numbers listed in Table 7-4, a two-octet field specifying the length of the TLV, and a variable field whose format is determined by the type.

**Table 7-4** *Type/Length/Value (TLV) types.*

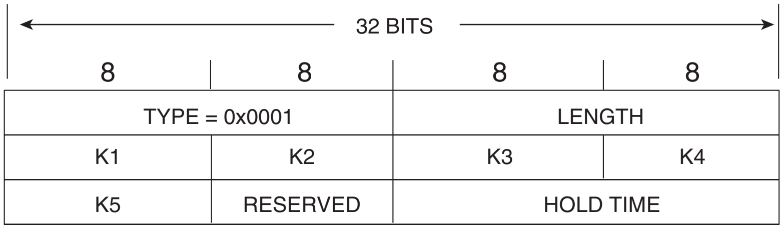
| Number | TLV Type |
|---|---|
| *General TLV Types* | |
| 0x0001 | EIGRP Parameters |
| 0x0003 | Sequence |
| 0x0004 | Software Version* |
| 0x0005 | Next Multicast Sequence |
| *IP-Specific TLV Types* | |
| 0x0102 | IP Internal Routes |
| 0x0103 | IP External Routes |
| *AppleTalk-Specific TLV Types* | |
| 0x0202 | AppleTalk Internal Routes |
| 0X0203 | AppleTalk External Routes |
| 0x0204 | AppleTalk Cable Configuration |
| *IPX-Specific TLV Types* | |
| 0x0302 | IPX Internal Routes |
| 0x0303 | IPX External Routes |

*This packet indicates whether the older software release is running (software version 0) or the newer release, as of IOS 10.3(11), 11.0(8), and 11.1(3), is running (version 1).

## General TLV Fields

These TLVs carry EIGRP management information and are not specific to any one routed protocol. The Parameters TLV, which is used to convey metric weights and the hold time, is shown in Figure 7-24. The Sequence, Software Version, and Next Multicast Sequence TLVs are used by the Cisco proprietary Reliable Multicast algorithm and are beyond the scope of this book.

**Figure 7-24** *EIGRP Parameters TLV.*



<table>
<tr><th colspan="4">32 BITS</th></tr>
<tr><th>8</th><th>8</th><th>8</th><th>8</th></tr>
<tr><td colspan="2">TYPE = 0x0001</td><td colspan="2">LENGTH</td></tr>
<tr><td>K1</td><td>K2</td><td>K3</td><td>K4</td></tr>
<tr><td>K5</td><td>RESERVED</td><td colspan="2">HOLD TIME</td></tr>
</table>

## IP-Specific TLV Fields

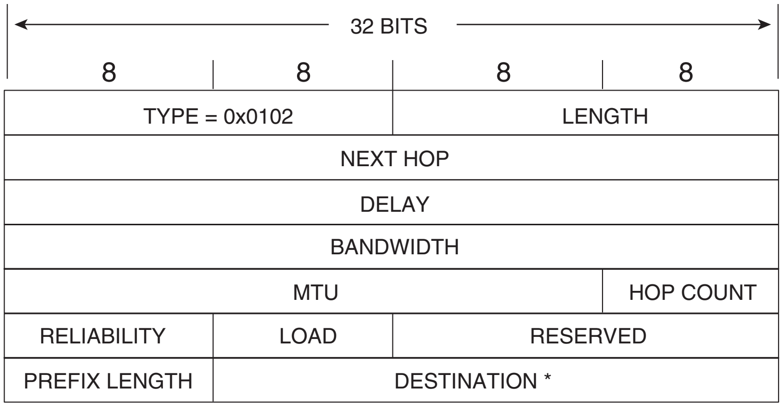
Each Internal and External Routes TLV contains one route entry. Every Update, Query, and Reply packet contains at least one Routes TLV.

The Internal and External Routes TLVs include metric information for the route. As noted earlier, the metrics used by EIGRP are the same metrics used by IGRP, although scaled by 256.

### IP Internal Routes TLV

An internal route is a path to a destination within the EIGRP autonomous system. The format of the Internal Routes TLV is shown in Figure 7-25.

**Figure 7-25** *The IP Internal Routes TLV.*



<table>
<tr><th colspan="4">32 BITS</th></tr>
<tr><th>8</th><th>8</th><th>8</th><th>8</th></tr>
<tr><td colspan="2">TYPE = 0x0102</td><td colspan="2">LENGTH</td></tr>
<tr><td colspan="4">NEXT HOP</td></tr>
<tr><td colspan="4">DELAY</td></tr>
<tr><td colspan="4">BANDWIDTH</td></tr>
<tr><td colspan="3">MTU</td><td>HOP COUNT</td></tr>
<tr><td>RELIABILITY</td><td>LOAD</td><td colspan="2">RESERVED</td></tr>
<tr><td>PREFIX LENGTH</td><td colspan="3">DESTINATION *</td></tr>
</table>

* This field is variable. If it is less than or more than 3 octets, the TLV will be padded with zeros to the next 4-octet boundary. For example, if the destination address is 10.1, the Destination field will be 2 octets and will be followed with a pad of 0x00. If the address is 192.168.16.64, the Destination field will be 4 octets and will be followed with a pad of 0x000000.

- **Next Hop** is the next-hop IP address. This address might or might not be the address of the originating router.
- **Delay** is the sum of the configured delays expressed in units of 10 microseconds. Notice that unlike the 24-bit delay field of the IGRP packet, this field is 32 bits. This larger field accommodates the 256 multiplier used by EIGRP. A delay of 0xFFFFFFFF indicates an unreachable route.
- **Bandwidth** is 256 × $BW_{IGRP(min)}$, or 2,560,000,000 divided by the lowest configured bandwidth of any interface along the route. Like Delay, this field is also eight bits larger than the IGRP field.
- **MTU** is the smallest Maximum Transmission Unit of any link along the route to the destination. Although an included parameter, it has never been used in the calculation of metrics.
- **Hop Count** is a number between 0x01 and 0xFF indicating the number of hops to the destination. A router will advertise a directly connected network with a hop count of 0; subsequent routers will record and advertise the route relative to the next-hop router.
- **Reliability** is a number between 0x01 and 0xFF that reflects the total outgoing error rates of the interfaces along the route, calculated on a five-minute exponentially weighted average. 0xFF indicates a 100 percent reliable link.
- **Load** is also a number between 0x01 and 0xFF, reflecting the total outgoing load of the interfaces along the route, calculated on a five-minute exponentially weighted average. 0x01 indicates a minimally loaded link.
- **Reserved** is an unused field and is always 0x0000.
- **Prefix Length** specifies the number of network bits of the address mask. *Destination* is the destination address of the route. Although the field is shown as a three-octet field in Figure 7-25 and Figure 7-26 (see next section), the field varies with the specific address. For example, if the route is to 10.1.0.0/16, the prefix length will be 16 and the destination will be a two-octet field containing 10.1. If the route is to 192.168.17.64/27, the prefix length will be 27 and the destination will be a four-octet field containing 192.168.16.64. If this field is not exactly three octets, the TLV will be padded with zeros to make it end on a four-octet boundary.

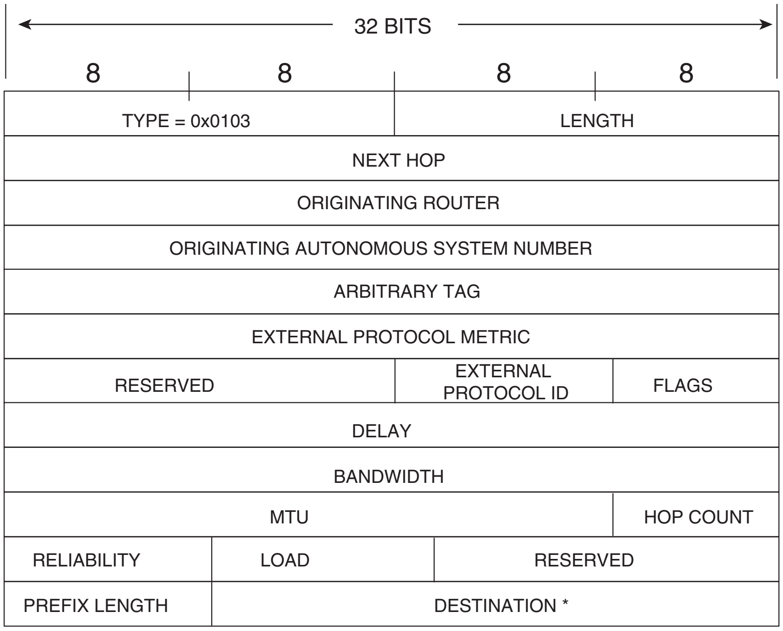
### IP External Routes TLV

An external route is a path that leads to a destination outside of the EIGRP autonomous system and that has been redistributed into the EIGRP domain. Figure 7-26 shows the format of the External Routes TLV:

- **Next Hop** is the next-hop IP address. On a multiaccess network, the router advertising the route might not be the best next-hop router to the destination. For example, an EIGRP-speaking router on an Ethernet link might also be speaking BGP and might be advertising a BGP-learned route into the EIGRP autonomous system. Because other routers on the link do not speak BGP, they might have no way of knowing that the interface to the BGP speaker is the best next-hop address. The Next Hop field allows the "bilingual" router to tell its EIGRP neighbors, "Use address A.B.C.D as the next hop instead of using my interface address."

- **Originating Router** is the IP address or router ID of the router that redistributed the external route into the EIGRP autonomous system.
- **Originating Autonomous System Number** is the autonomous system number of the router originating the route.
- **Arbitrary Tag** may be used to carry a tag set by route maps. See Chapter 14 for information on he use of route maps.
- **External Protocol Metric** is, as the name implies, the metric of the external protocol. This field is used, when distributing with IGRP, to track the IGRP metric.
- **Reserved** is an unused field and is always 0x0000.
- **External Protocol ID** specifies the protocol from which the external route was learned. Table 7-5 lists the possible values of this field.

**Figure 7-26** *The IP External Routes TLV.*



* This field is variable. If it is less than or more than 3 octets, the TLV will be padded with zeros to the next 4-octet boundary. For example, if the destination address is 10.1, the Destination field will be 2 octets and will be followed with a pad of 0x00. If the address is 192.168.16.64, the Destination field will be 4 octets and will be followed with a pad of 0x000000.

**Table 7-5** *Values of the External Protocol ID field.*

| Code | External Protocol |
|---|---|
| 0x01 | IGRP |
| 0x02 | EIGRP |
| 0x03 | Static Route |
| 0x04 | RIP |
| 0x05 | Hello |
| 0x06 | OSPF |
| 0x07 | IS-IS |
| 0x08 | EGP |
| 0x09 | BGP |
| 0x0A | IDRP |
| 0x0B | Connected Link |

- **Flags** currently constitute just two flags. If the right-most bit of the eight-bit field is set (0x01), the route is an external route. If the second bit is set (0x02), the route is a candidate default route. Default routes are discussed in Chapter 12.

The remaining fields describe the metrics and the destination address. The descriptions of these fields are the same as those given in the discussion of the Internal Routes TLV.

## Address Aggregation

Chapter 1, "Routing Basics," introduced the practice of subnetting, in which the address mask is extended into the host space to address multiple data links under one major network address. Chapter 6 introduced the practice of variable-length subnet masking, in which the address mask is extended even more to create subnets within subnets.

From an opposite perspective, a subnet address may be thought of as a summarization of a group of sub-subnets. And a major network address may be thought of as a summarization of a group of subnet addresses. In each case, the summarization is achieved by reducing the length of the address mask.

Address aggregation takes summarization a step further by breaking the class limits of major network addresses. An aggregate address represents a numerically contiguous group of network addresses, known as a supernet.[13] Figure 7-27 shows an example of an aggregate address.

[13] An aggregate is, more correctly, any summarized group of addresses. For clarity, in this book the term aggregate refers to a summarization of a group of major network addresses.

**Figure 7-27** *This group of network addresses can be represented with a single aggregate address, or subnet.*

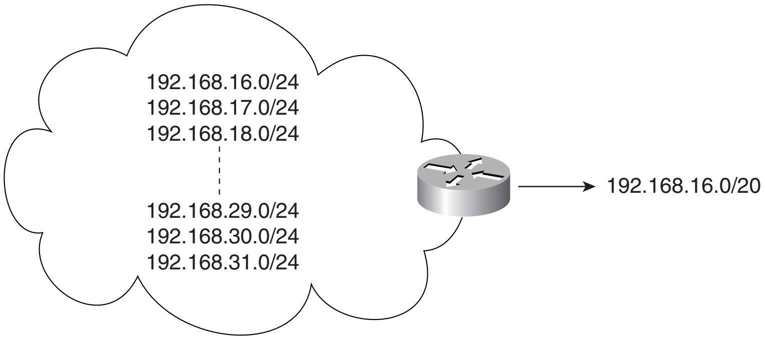


Figure 7-28 shows how the aggregate address of Figure 7-27 is derived. For a group of network addresses, find the bits that are common to all of the addresses and mask these bits. The masked portion is the aggregate address.

**Figure 7-28** *The aggregate address is derived by masking all the common bits of a group of numerically contiguous network addresses.*

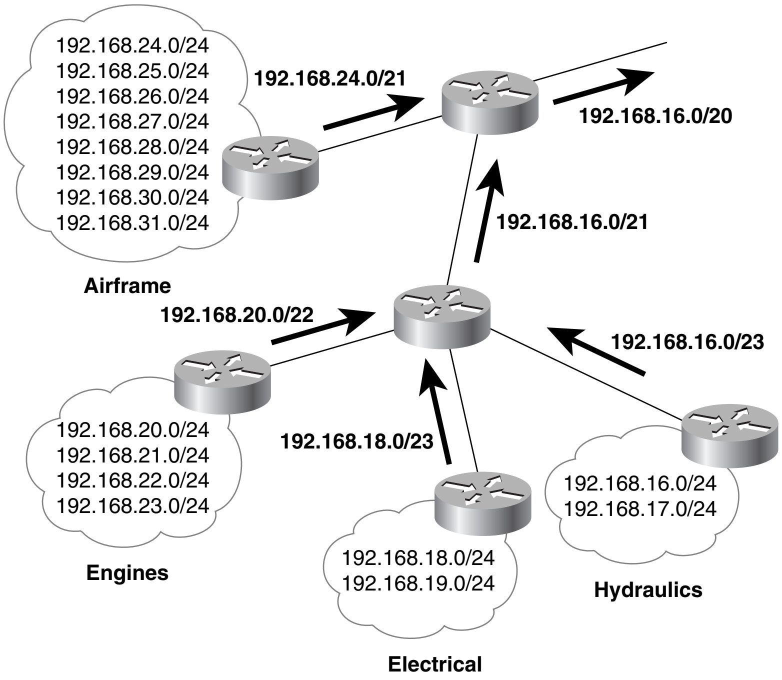
```
11111111111111111111111100000000 = 24-bit mask
11000000101010000001000000000000 = 192.168.16.0/24
11000000101010000001000100000000 = 192.168.17.0/24
11000000101010000001001000000000 = 192.168.18.0/24
11000000101010000001001100000000 = 192.168.19.0/24
11000000101010000001010000000000 = 192.168.20.0/24
11000000101010000001010100000000 = 192.168.21.0/24
11000000101010000001011000000000 = 192.168.22.0/24
11000000101010000001011100000000 = 192.168.23.0/24
11000000101010000001100000000000 = 192.168.24.0/24
11000000101010000001100100000000 = 192.168.25.0/24
11000000101010000001101000000000 = 192.168.26.0/24
11000000101010000001101100000000 = 192.168.27.0/24
11000000101010000001110000000000 = 192.168.28.0/24
11000000101010000001110100000000 = 192.168.29.0/24
11000000101010000001111000000000 = 192.168.30.0/24
11000000101010000001111100000000 = 192.168.31.0/24
--------------------------------------------------
11000000101010000001000000000000 = 192.168.16.0/20
```

When designing a supernet, it is important that the member addresses should form a complete, contiguous set of the formerly masked bits. In Figure 7-28, for example, the 20-bit mask of the aggregate address is four bits less than the mask of the member addresses. Of the four "difference" bits, notice that they include every possible bit combination from 0000 to 1111. Failure to follow this design rule could make the addressing scheme confusing, could reduce the efficiency of aggregate routes, and might lead to routing loops and black holes.

The obvious advantage of summary addressing is the conservation of network resources. Bandwidth is conserved by advertising fewer routes, and CPU cycles are conserved by processing fewer routes. Most important, memory is conserved by reducing the size of the route tables.

Classless routing, VLSM, and aggregate addressing together provide the means to maximize resource conservation by building address hierarchies. Unlike IGRP, EIGRP supports all of these addressing strategies. In Figure 7-29, the engineering division of Treetop Aviation has been assigned 16 class C addresses. These addresses have been assigned to the various departments according to need.

**Figure 7-29** *At Treetop Aviation, several departments within a larger division are aggregating addresses. In turn, the entire division can be advertised with a single aggregate address (192.168.16.0/20).*

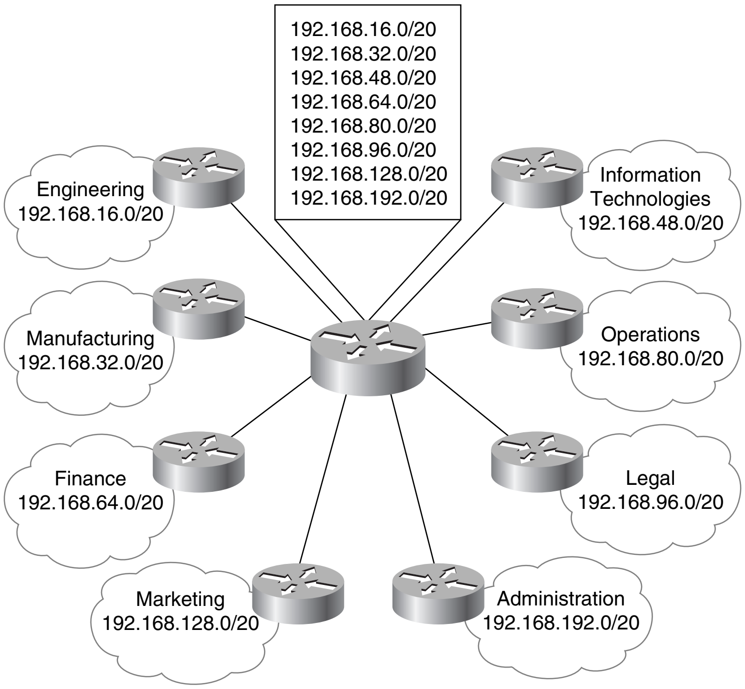


The aggregate addresses of the engines, electrical, and hydraulics departments are themselves aggregated into a single address, 192.168.16.0/21. That address and the aggregate address of the airframe department are aggregated into the single address 192.168.16.0/20, which represents the entire engineering division.

Other divisions may be similarly represented. For example, if Treetop Aviation had a total of eight divisions and if those divisions were all addressed similarly to the engineering division, the backbone router at the top of the hierarchy might have only eight routes (Figure 7-30).

The hierarchical design is continued within each department of each division by subnetting the individual network addresses. VLSM may be used to further divide the subnets. The routing protocols will automatically summarize the subnets at network boundaries, as discussed in previous chapters.

**Figure 7-30** *Although there are 128 major network addresses and possibly more than 32,000 hosts in this network, the backbone router has only eight aggregate addresses in its route table.*



Address aggregation also allows both address conservation and address hierarchies in the Internet. With the exponential growth of the Internet, two increasing concerns have been the depletion of available IP addresses (particularly class B addresses) and the huge databases needed to store the Internet routing information.

A solution to this problem is *Classless Interdomain Routing* (CIDR).[14] Under CIDR, aggregates of class C addresses are allocated by the IANA to the various worldwide address assignment authorities such as Asia Pacific Network Information Centre (APNIC) in Asia, American Registry for Internet Numbers (ARIN) in North America, and Réseaux IP Européens (RIPE) in Europe. The aggregates are organized geographically, as shown in Table 7-6.

**Table 7-6** *CIDR address allocations by geographic region.*

| Region | Address Range |
|---|---|
| Multiregional | 192.0.0.0–193.255.255.255 |
| Europe | 194.0.0.0–195.255.255.255 |
| Others | 196.0.0.0–197.255.255.255 |

[14] V. Fuller, T. Li, J. I. Yu, and K. Varadhan. “Classless Inter-Domain Routing (CIDR): An Address Assignment and Aggregation Strategy,” RFC 1519. September 1993.

**Table 7-6** *CIDR address allocations by geographic region. (Continued)*

| Region | Address Range |
|---|---|
| North America | 198.0.0.0–199.255.255.255 |
| Central/South America | 200.0.0.0–201.255.255.255 |
| Pacific Rim | 202.0.0.0–203.255.255.255 |
| Others | 204.0.0.0–205.255.255.255 |
| Others | 206.0.0.0–207.255.255.255 |

The address assignment authorities in turn divide their portions among the regional Internet Service Providers (ISPs). When an organization applies for an IP address and requires addressing for fewer than 32 subnets and 4096 hosts, it will be given a contiguous group of class C addresses called a *CIDR block*.

In this way, the Internet routers of individual organizations might advertise a single summary address to their ISP. The ISP, in turn, aggregates all of its addresses, and ideally all ISPs in a region of the world may be summarized by the addresses of Table 7-6. Knowing that the current size of the Internet routing table is approaching 200,000 routes, it is obvious that CIDR has not been adhered to as well as intended. Nonetheless, the concept is a good one. Similar efforts are under way to constrain IPv6 address assignments geographically; we can only hope that the efforts are more successful than they were with IPv4.

## EIGRP and IPv6

As the second edition of this book is being written, EIGRP does not support IPv6. However, an effort to extend EIGRP for IPv6 support is under way, and it is expected that by the time you are reading this chapter that extension will be available. Because EIGRP packets use TLVs to carry data, extending the protocol to support IPv6 will be a simple matter of adding IPv6-specific TLVs.

# Configuring EIGRP

The case studies in this section demonstrate a basic EIGRP configuration and then examine summarization techniques and interoperability with IGRP.

## Case Study: A Basic EIGRP Configuration

EIGRP requires only two steps to begin the routing process:

**Step 1** Enable EIGRP with the command **router eigrp** *process-id*.

**Step 2** Specify each major network on which to run EIGRP with the **network** command.

The process ID may be any number between 1 and 65535 (0 is not allowed), and it may be arbitrarily chosen by the network administrator, if it is the same for all EIGRP processes in all routers that must share information. Alternatively, the number might be a publicly assigned autonomous system number. Figure 7-31 shows a simple network; the configurations for the three routers are displayed in Example 7-14, Example 7-15, and Example 7-16.

**Example 7-14** *Earhart.*

```
router eigrp 15
 network 172.20.0.0
```

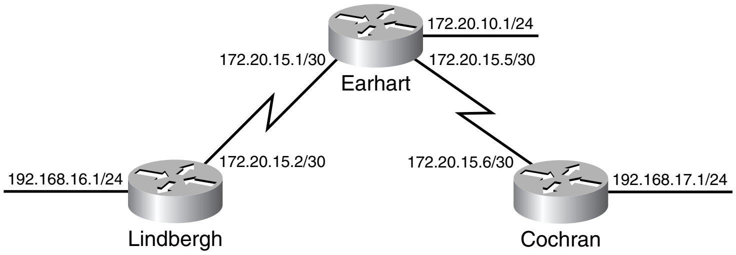
**Example 7-15** *Cochran.*

```
router eigrp 15
 network 172.20.0.0
 network 192.167.17.0
```

**Example 7-16** *Lindbergh.*

```
router eigrp 15
 network 172.20.0.0
 network 192.167.16.0
```

**Figure 7-31** *Unlike IGRP, EIGRP will support the VLSM requirements of this network.*



Earhart's route table is shown in Example 7-17. The table shows that the default EIGRP administrative distance is 90 and that network 172.20.0.0 is variably subnetted.

**Example 7-17** *Earhart's route table.*

```
Earhart#show ip route
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, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route
Gateway of last resort is not set
     172.20.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
D    192.168.17.0/24 [90/2195456] via 172.20.15.6, 00:00:01, Serial0.1
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:01, Serial0.2
Earhart#
```

The network of Figure 7-31 uses default metrics, unlike the earlier examples in this chapter, so a review of the EIGRP metric calculation in a more realistic scenario might be useful.

Tracing the route from Earhart to network 192.168.16.0, the path traverses a serial interface and an Ethernet interface, each with default metric values. The minimum bandwidth of the route will be that of the serial interface,[15] and the delay will be the sum of the two interface delays. Referring back to Table 7-1:

$$BW_{EIGRP(min)} = 256 \times 6476 = 1657856$$
$$DLY_{EIGRP(sum)} = 256 \times (2000 + 100) = 537600$$

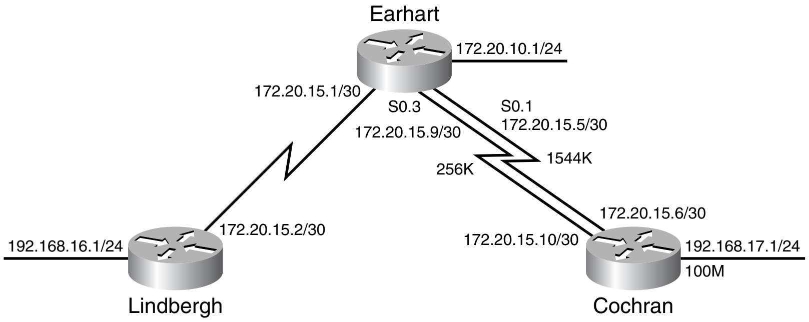
Therefore,

$$Metric = 1657856 + 537600 = 2195456$$

## Case Study: Unequal-Cost Load Balancing

Given up to 16 parallel routes of equal cost,[16] EIGRP will do equal-cost load balancing under the same CEF/fast/process switching constraints as RIP. Unlike RIP, EIGRP can also perform unequal-cost load balancing. An additional serial link has been added between Earhart and Cochran in Figure 7-32, with a configured bandwidth of 256K. The goal is to have Earhart perform unequal-cost load balancing across these two links—spreading the traffic load inversely proportional to the metrics of the link.

**Figure 7-32** *EIGRP can be configured to perform unequal-cost load balancing across links such as the two between Earhart and Cochran.*



Examining the route from Earhart's S0.1 interface to network 192.168.17.0, the minimum bandwidth is 1544K (assuming Cochran's Ethernet interface is using the default 100000K bandwidth for Fast Ethernet). Referring to Table 7-1, $DLY_{EIGRP(sum)}$ for the serial interface and the Fast Ethernet interface is $256 \times ( 2000 + 10 ) = 514560$. $BW_{EIGRP(min)}$ is $256 \times (10^7/1544) = 1657856$, so the composite metric of the route is 514560 + 1657856) = 2172416.

[15] Remember that the default bandwidth of a serial interface is 1544K.

[16] The default is four paths. See the case study on setting maximum paths for further details.

The minimum bandwidth on the route via Earhart's S0.3 to 192.168.17.0 is 256K; $DLY_{EIGRP(sum)}$ is the same as on the first route. Therefore, the composite metric for this route is $256 \times (10^7/256) + 514560 = 10514432$. Without further configuration, EIGRP will simply select the path with the lowest metric cost. Example 7-18 shows that Earhart is using only the path via Serial 0.1, with a metric of 2172416.

**Example 7-18** *Earhart is using only the lowest-cost link to network 192.168.17.0. Additional configuration is needed to enable unequal-cost load balancing.*

```
Earhart#show ip route
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, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o – ODR
       P - periodic downloaded static route
Gateway of last resort is not set
     172.20.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.20.10.0/27 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2172416] via 172.20.15.6, 00:00:09, Serial0.1
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:09, Serial0.2
Earhart#
```

The **variance** command is used to determine which routes are feasible for unequal-cost load sharing. Variance defines a multiplier by which a metric may differ, or vary, from the metric of the lowest-cost route. Any route whose metric exceeds the metric of the lowest-cost route, multiplied by the variance, will not be considered a feasible route.

The default variance is one, meaning that the metrics of multiple routes must be equal, to load balance. Variance must be specified in whole numbers.

The metric of Earhart's route through S0.3 is 10514432/2172416 = 4.8 times larger than the metric of the S0.1 route. So to conduct unequal-cost load balancing over both links, the variance at Earhart should be five. The EIGRP configuration is in Example 7-19.

**Example 7-19** *Earhart's configuration uses a variance of five to perform unequal-cost load sharing using EIGRP.*

```
router eigrp 15
 network 172.20.0.0
 variance 5
```

After specifying a variance of five at Earhart, its route table will include the second, higher-cost route (Example 7-20). The following three conditions must be met for a route to be included in unequal-cost load sharing:

- The maximum-paths limit must not be exceeded as a result of adding the route to a load-sharing "group."

- The next-hop router must be metrically closer to the destination. That is, its metric for the route must be smaller than the local router's metric. A next-hop router, being closer to the destination, is often referred to as the *downstream* router.
- The metric of the lowest-cost route, when multiplied by the variance, must be greater than the metric of the route to be added.

**Example 7-20** *The composite metric of the second path to 192.168.17.0 from Earhart is 10514432, or 4.8 times the metric of the lowest-cost route. EIGRP will enter the second path into the route table if the variance is set to at least five.*

```
Earhart(config)#router eigrp 15
Earhart(config-router)#variance 5
Earhart(config-router)#^Z
Earhart#clear ip route *
Earhart#show ip route
Gateway of last resort is not set
     172.20.0.0/16 is variably subnetted, 4 subnets, 2 masks
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2172416] via 172.20.15.6, 00:00:02, Serial0.1
                     [90/10514432] via 172.20.15.10, 00:00:02, Serial0.3
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:02, Serial0.2
Earhart#
```

The rules concerning per destination versus per packet load sharing, discussed in Chapter 3, "Static Routing," apply here as well. Load sharing is per destination if the packet is fast switched or CEF switched using the default CEF configuration, and per packet if process switching is used or if the CEF configuration was modified. Example 7-21 shows a debug output resulting from 20 ping packets being sent through Earhart; CEF and fast switching have been turned off with **no ip cef** and **no ip route-cache**, and the router is performing unequal-cost, per packet load balancing. For every five packets sent over the 1544K link (to next hop 172.20.15.6), one packet is sent over the 256K link (to next hop 172.20.15.10). This corresponds to the approximately five-to-one variance of the metrics of these two paths.

**Example 7-21** *Per packet load sharing is being performed, with one packet being sent over the high-cost link for every five packets sent over the low-cost link.*

```
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
```

*continues*

**Example 7-21** *Per packet load sharing is being performed, with one packet being sent over the high-cost link for every five packets sent over the low-cost link. (Continued)*

```
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.3), g=172.20.15.10,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.3), g=172.20.15.10,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.1), g=172.20.15.6,
    len 100, forward
IP: s=172.20.15.2 (Serial 2), d=192.168.17.1 (Serial0.3), g=172.20.15.10,
    len 100, forward
Earhart#
```

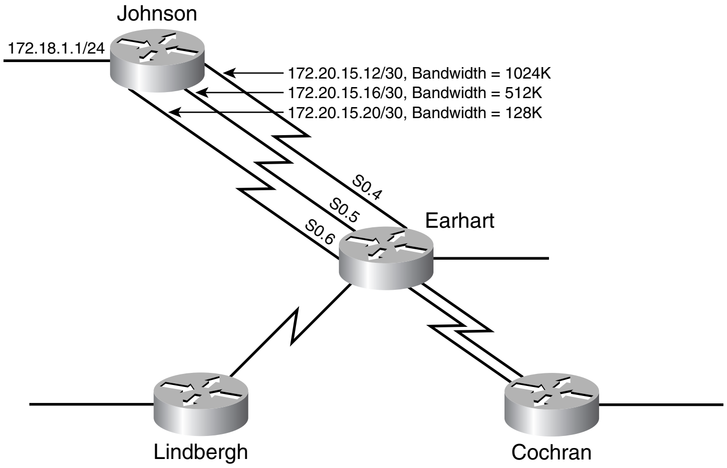
If variance is set at one, EIGRP enters only the lowest-cost route to a destination into the route table. In some situations, however—for example, to decrease reconvergence time or aid in troubleshooting—all feasible routes should be entered into the table, even though no load balancing should occur. All packets should use the lowest-cost route and switch to the next-best path only if the primary fails. There is an implicit default command (that is, it exists, but is not observed in the configuration file) of **traffic-share balanced**. To configure the router to only use the minimum-cost path even when multiple paths are shown in the route table, change this default to **traffic-share min**. If there are multiple minimum-cost paths and **traffic-share min** is configured, EIGRP will perform equal-cost load balancing.

## Case Study: Setting Maximum Paths

The maximum number of routes over which EIGRP can load balance is set with the **maximum-paths** *paths* command. *paths* may be any number from 1 to 16 in IOS 12.3(2)T and later 12.3(T) releases and any number from 1 to 6 in earlier versions. The default for all versions is 4.

Figure 7-33 shows three parallel paths of varying costs from Earhart to address 172.18.0.0. The network administrator wants to load balance over a maximum of only two of these routes while ensuring that if either of these paths should fail, the third route will replace it.

**Figure 7-33** *The maximum-paths and variance commands can be used together to configure load balancing over only two of the three links between Earhart and Johnson. If either link fails, the third will take its place.*


Johnson
172.18.1.1/24
172.20.15.12/30, Bandwidth = 1024K
172.20.15.16/30, Bandwidth = 512K
172.20.15.20/30, Bandwidth = 128K
S0.4
S0.5
S0.6
Earhart
Lindbergh
Cochran

The metrics from Earhart are

Via S0.4: 256 × (9765 + (2000 + 10)) = 3014400
Via S0.5: 256 × (19531 + (2000 + 10)) = 5514496
Via S0.6: 256 × (78125 + (2000 + 10)) = 20514560

The metric of the S0.6 route is 6.8 times as large as the lowest-cost metric, so the variance is seven.

Earhart's EIGRP configuration is displayed in Example 7-22.

**Example 7-22** *Earhart's configuration uses the variance command and the maximum-paths command to provide unequal-cost load sharing over a specified maximum number of paths.*

```
router eigrp 15
 variance 7
 network 172.20.0.0
 maximum-paths 2
```

The **variance** command ensures that any of the three routes to 172.18.0.0 is feasible; the **maximum-paths** command limits the load-sharing group to only the two best routes. The results of this configuration can be seen in Example 7-23. The first route table shows that Earhart was load balancing over the two links with the lowest of the three metrics, S0.4 and S0.5. After a failure of the S0.4 link, the second route table shows that the router is now load balancing over the S0.5 and S0.6 links. In each instance, the router will load balance inversely proportional to the metrics of the two paths.

**Example 7-23** *The route table for Earhart, before and after the failure of one of three links, shows the results of using the variance and maximum-paths commands to configure load sharing to 172.18.0.0.*

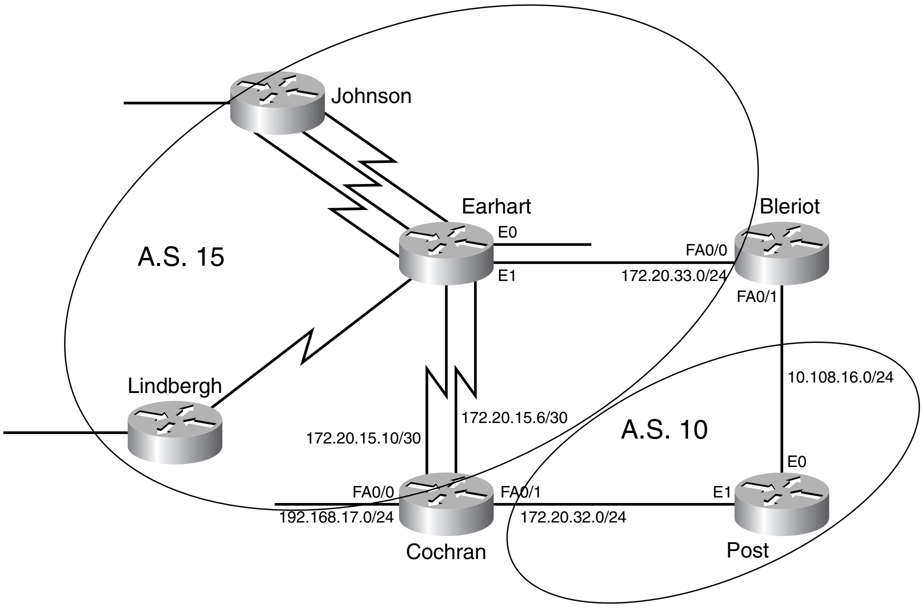
```
Earhart#debug eigrp neighbor
EIGRP Neighbors debugging is on
Earhart#show ip route
Gateway of last resort is not set

D    172.18.0.0/16 [90/5514496] via 172.20.15.18, 00:00:16, Serial0.5
                   [90/3014400] via 172.20.15.14, 00:00:16, Serial0.4
     172.20.0.0/16 is variably subnetted, 7 subnets, 2 masks
C       172.20.15.20/30 is directly connected, Serial0.6
C       172.20.15.16/30 is directly connected, Serial0.5
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.12/30 is directly connected, Serial0.4
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2195456] via 172.20.15.6, 00:00:18, Serial0.1
                     [90/10537472] via 172.20.15.10, 00:00:18, Serial0.3
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:19, Serial0.2
Earhart#
1w6d: EIGRP: Holdtime expired
1w6d: EIGRP: Neighbor 172.20.15.14 went down on Serial0.4
Earhart#show ip route
Gateway of last resort is not set
D    172.18.0.0/16 [90/5514496] via 172.20.15.18, 00:00:09, Serial0.5
                   [90/20514560] via 172.20.15.22, 00:00:09, Serial0.6
     172.20.0.0/16 is variably subnetted, 7 subnets, 2 masks
C       172.20.15.20/30 is directly connected, Serial0.6
C       172.20.15.16/30 is directly connected, Serial0.5
C       172.20.10.0/24 is directly connected, Ethernet0
C       172.20.15.4/30 is directly connected, Serial0.1
C       172.20.15.0/30 is directly connected, Serial0.2
C       172.20.15.12/30 is directly connected, Serial0.4
C       172.20.15.8/30 is directly connected, Serial0.3
D    192.168.17.0/24 [90/2195456] via 172.20.15.6, 00:00:26, Serial0.1
                     [90/10537472] via 172.20.15.10, 00:00:26, Serial0.3
D    192.168.16.0/24 [90/2195456] via 172.20.15.2, 00:00:26, Serial0.2
Earhart#
```

Care should be taken when configuring multiple parallel paths in a network. Too many parallel paths can increase the amount of time it takes for EIGRP to converge when a link fails, because the number of neighboring routers has increased, therefore increasing the querying scope.

## Case Study: Multiple EIGRP Processes

Two new routers, Bleriot and Post, have been added to the network. A decision has been made to create two EIGRP process domains in the network with no communications between the two. Figure 7-34 shows the two autonomous systems and the related links for each.

**Figure 7-34** *The routers Bleriot and Cochran will each run multiple EIGRP processes to facilitate the creation of separate autonomous systems (AS 10 and AS 15) within this IGP.*



The configurations for Post, Johnson, Lindbergh, and Earhart are straightforward: Johnson, Earhart, and Lindbergh will run EIGRP 15, and Post will run EIGRP 10. At Bleriot, the configuration will be as displayed in Example 7-24.

**Example 7-24** *Bleriot's configuration for EIGRP 15 and EIGRP 10.*

```
router eigrp 15
 network 172.20.0.0
!
router eigrp 10
 network 10.0.0.0
```

Each process will run only on the interfaces of the networks specified. At Cochran, all interfaces other then FA0/0 belong to network 172.20.0.0 (see Example 7-25).

Using the **passive-interface** command prevents EIGRP Hellos from being sent on data links where they don't belong. Note that because Cochran's interfaces are in network 172.20.0.0, the **passive-interface** command is used to restrict unnecessary routing protocol traffic. For EIGRP, this command blocks unnecessary Hellos. No adjacencies will be formed; therefore, no other EIGRP traffic will be sent.

**Example 7-25** *Cochran's EIGRP 15 and EIGRP 10 configuration requires passive interfaces because the same major network number is used on both of Cochran's connected interfaces.*

```
router eigrp 15
 passive-interface Fastethernet0/1
 network 172.20.0.0
!
router eigrp 10
 passive-interface Serial0/0.1
 passive-interface Serial 0/0.2
 network 172.20.0.0
```

Bleriot's neighbor table (Example 7-26) shows that there is one neighbor for EIGRP process 10, 10.108.16.2, and one neighbor for process 15, 172.20.33.1.

**Example 7-26** *The neighbors associated with each of the multiple EIGRP processes are displayed using the* **show ip eigrp neighbor** *command.*

```
Bleriot#show ip eigrp neighbors
IP-EIGRP neighbors for process 15
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   172.20.33.1             Fa0/0             11 00:31:19   34   204  0  44
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   10.108.16.2             Fa0/1             10 00:19:21  289  1734  0  6
Bleriot#
```

In lieu of passive interfaces, the network statement for EIGRP can be configured with wildcard bits. The wildcard bits specify which bits of the address are to be used when identifying interfaces to include in the EIGRP process.

Cochran's configuration is modified to that shown in Example 7-27.

**Example 7-27** *Cochran's EIGRP 15 and EIGRP 10 configuration uses wildcard bits with the network to narrow down the interfaces that will run EIGRP for a given process.*

```
router eigrp 15
 network 172.20.15.0 0.0.0.255
!
router eigrp 10
 network 172.20.32.0 0.0.0.255
```

Cochran's configuration in Example 7-27 specifies that interfaces with addresses with the first three octets equal to 172.20.15 run EIGRP process 15, and interfaces with the first three octets equal to 172.20.32 run EIGRP process 10.

## Case Study: Disabling Automatic Summarization

By default, EIGRP summarizes at network boundaries as do the protocols covered in previous chapters. Unlike those protocols, however, EIGRP's automatic summarization can be disabled.

Look at Bleriot's route table in Example 7-28. Notice that the entry for address 172.20.32.0 is known via Fast Ethernet 0/0, even though that path goes through one Fast Ethernet link and one serial link, from Bleriot, through Earhart to Cochran. The path via Post, attached to Fast Ethernet 0/1, is only one Fast Ethernet hop away. Look more closely at the route table, and you'll see the only entry for an address other then 10.108.16.0 known via Fast Ethernet 0/1 is 172.20.0.0/16. Post has summarized the 172.20.32.0 address before it advertises it to Bleriot over the 10.0.0.0 network boundary. To enable Bleriot to forward traffic to 172.20.32.0 via Post, disable automatic summarization on Post using the command **no auto-summary**.

**Example 7-28** *EIGRP automatic summarization can cause, in some cases, sub-optimal route choices.*

```
Bleriot#show ip route
Gateway of last resort is not set
D    172.18.0.0/16 [90/3016960] via 172.20.33.1, 00:26:42, FastEthernet0/0
     172.20.0.0/16 is variably subnetted, 10 subnets, 4 masks
D       172.20.32.0/24 [90/2174976] via 172.20.33.1, 00:14:46, FastEthernet0/0
C       172.20.33.0/24 is directly connected, FastEthernet0/0
D       172.20.15.20/30
           [90/20514560] via 172.20.33.1, 00:20:28, FastEthernet0/0
D       172.20.15.16/30
           [90/5514496] via 172.20.33.1, 00:20:28, FastEthernet0/0
D       172.20.10.0/27 [90/284160] via 172.20.33.1, 00:43:34, FastEthernet0/0
D       172.20.15.4/30 [90/2172416] via 172.20.33.1, 00:26:59, FastEthernet0/0
D       172.20.15.0/30 [90/2172416] via 172.20.33.1, 00:27:18, FastEthernet0/0
D       172.20.0.0/16 [90/284160] via 10.108.16.2, 00:14:50, FastEthernet0/1
D       172.20.15.12/30
           [90/3014400] via 172.20.33.1, 00:26:49, FastEthernet0/0
D       172.20.15.8/30
           [90/10514432] via 172.20.33.1, 00:20:47, FastEthernet0/0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.108.16.0 is directly connected, FastEthernet0/1
D    192.168.16.0/24 [90/2198017] via 172.20.33.1, 00:27:33, FastEthernet0/0
D    192.168.17.0/24 [90/2174976] via 172.20.33.1, 00:00:38, FastEthernet0/0
Bleriot#
```

Post's configuration is displayed in Example 7-29.

**Example 7-29** *Post's configuration disables automatic summarization for EIGRP.*

```
router eigrp 10
 network 10.0.0.0
 network 172.20.0.0
 no auto-summary
```

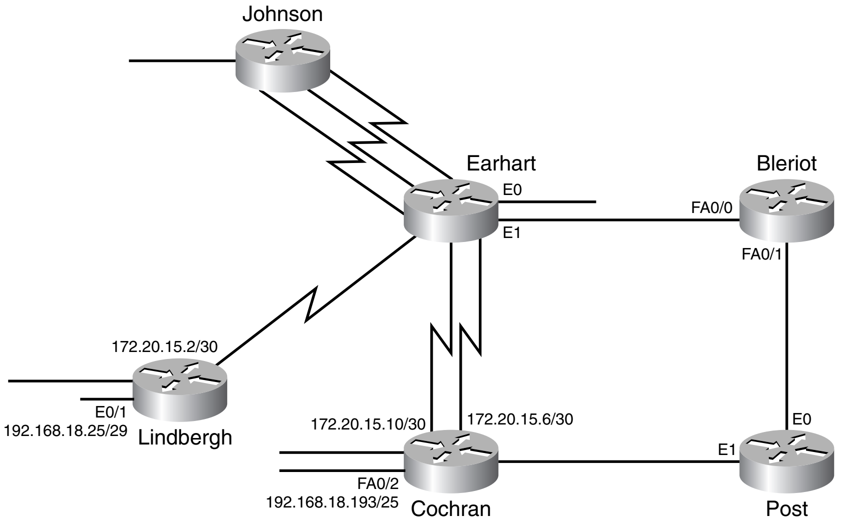
Bleriot's new route table is shown in Example 7-30.

**Example 7-30** *After disabling EIGRP automatic summarization, subnets of distant addresses can be seen in the route tables.*

```
Bleriot#show ip route
Gateway of last resort is not set
D    172.18.0.0/16 [90/3016960] via 172.20.33.1, 00:35:27, FastEthernet0/0
     172.20.0.0/16 is variably subnetted, 9 subnets, 3 masks
D       172.20.32.0/24 [90/284160] via 10.108.16.2, 00:00:55, FastEthernet0/1
C       172.20.33.0/24 is directly connected, FastEthernet0/0
D       172.20.15.20/30
           [90/20514560] via 172.20.33.1, 00:29:13, FastEthernet0/0
D       172.20.15.16/30
           [90/5514496] via 172.20.33.1, 00:29:13, FastEthernet0/0
D       172.20.10.0/27 [90/284160] via 172.20.33.1, 00:52:19, FastEthernet0/0
D       172.20.15.4/30 [90/2172416] via 172.20.33.1, 00:35:44, FastEthernet0/0
D       172.20.15.0/30 [90/2172416] via 172.20.33.1, 00:36:03, FastEthernet0/0
D       172.20.15.12/30
           [90/3014400] via 172.20.33.1, 00:35:34, FastEthernet0/0
D       172.20.15.8/30
           [90/10514432] via 172.20.33.1, 00:29:20, FastEthernet0/0
     10.0.0.0/24 is subnetted, 1 subnets
C       10.108.16.0 is directly connected, FastEthernet0/1
D    192.168.16.0/24 [90/2198016] via 172.20.33.1, 00:36:06, FastEthernet0/0
D    192.168.17.0/24 [90/2174976] via 172.20.33.1, 00:00:38, FastEthernet0/0
Bleriot#
```

Figure 7-35 shows another situation in which disabling summarization is useful.

**Figure 7-35** *Disabling automatic summarization at Cochran and Lindbergh prevents ambiguous routing to network 192.168.18.0.*



New Ethernet links have been added to routers Cochran and Lindbergh, and their addresses create a discontiguous subnet. The default behavior of both routers is to see themselves as border routers between major networks 172.20.0.0 and 192.168.18.0. As a result, Earhart will receive summary advertisements to 192.168.18.0 on its serial interfaces to both Lindbergh and Cochran. The result is an ambiguous routing situation in which Earhart records two equal-cost paths to 192.168.18.0; a packet destined for one of the subnets might or might not be routed to the correct link.

After disabling automatic summarization, Lindbergh's configuration will be as in Example 7-31.

**Example 7-31** *Lindbergh's configuration disables automatic summarization.*

```
router eigrp 15
 network 172.20.0.0
 network 192.168.16.0
 network 192.168.18.0
 no auto-summary
```

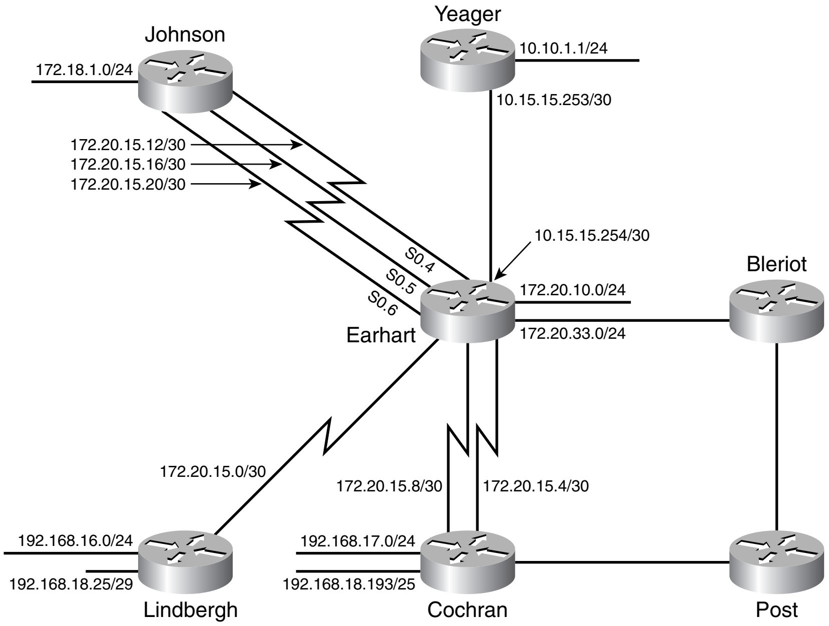
By turning off summarization at Lindbergh and Cochran, the individual subnets 192.168.18.24/29 and 192.168.18.128/25 will be advertised into network 172.20.0.0, eliminating the ambiguities at Earhart.

## Case Study: Stub Routing

Recall the discussion of DUAL earlier in this chapter. When an entry in a router's EIGRP topology table changes for the worse (either the metric increases, or the successor is no longer accessible), if there is no feasible successor for the address, the entry goes into Active state, and the router sends query packets to all its neighbors. If Earhart's link to Yeager, in Figure 7-36, goes down, Earhart sends queries to all its neighbors, including Johnson and Lindbergh, to find out if any neighbors have a path to Yeager. Earhart cannot modify its active entries in the topology table until it hears responses from all its queries regarding that entry. If a problem develops on the link to Lindbergh before Earhart has received a response to the query it sent about Yeager's addresses, Yeager's addresses will remain Active, even if the link between Earhart and Yeager comes back up.

Johnson and Lindbergh, in Figure 7-36, do not have back-door routes to any other site in the network. They are spoke routers in a hub-and-spoke design. The routers are not used to provide transit paths to any addresses in the network. When Lindbergh or Johnson need to forward a packet to an address that is not local to its site, the packet is forwarded to Earhart. Lindbergh knows of one path to 172.20.10.0, for instance, and that path is via Earhart. There is no need to send Johnson queries about addresses in other locations of the network and risk causing network instabilities. Johnson and Lindbergh can be configured with stub routing.

**Figure 7-36** *Yeager is added to the network with a single link to Earhart.*



A router that has EIGRP Stub neighbors will not send queries to the stubs, thereby eliminating the chance that a stub-configured remote site will cause stuck in active conditions, and routing instabilities in other parts of the network.

Johnson is configured as an EIGRP stub router. Johnson's stub router configuration is displayed in Example 7-32.

**Example 7-32** *Johnson's EIGRP stub router configuration.*

```
router eigrp 15
 eigrp stub
```

No configuration changes are required on Earhart, the hub router.

The command **eigrp stub** causes Johnson to send updates containing its connected and summary routes only. Johnson can be configured to include any combination of connected routes, summary routes, static routes, or routes that have been redistributed into EIGRP, with the command:

```
eigrp stub {connected | redistributed | static | summary | receive-only}
```

Johnson can also be configured to not send any route information in updates, with the **receive-only** option. With the **receive-only** option, the remote router will not include any addresses in an update. Addresses connected to the Johnson router would have to be advertised to the rest of the network in some other way to ensure that traffic can reach the site, perhaps with static routes configured on Earhart.

To verify a neighbor is configured as a stub router, use the command **show ip eigrp neighbor detail** on the hub router, as shown in Example 7-33. Earhart's output shows that Johnson is configured as a stub.

**Example 7-33** **show ip eigrp neighbor detail** *displays which neighbor routers are configured as EIGRP stubs.*

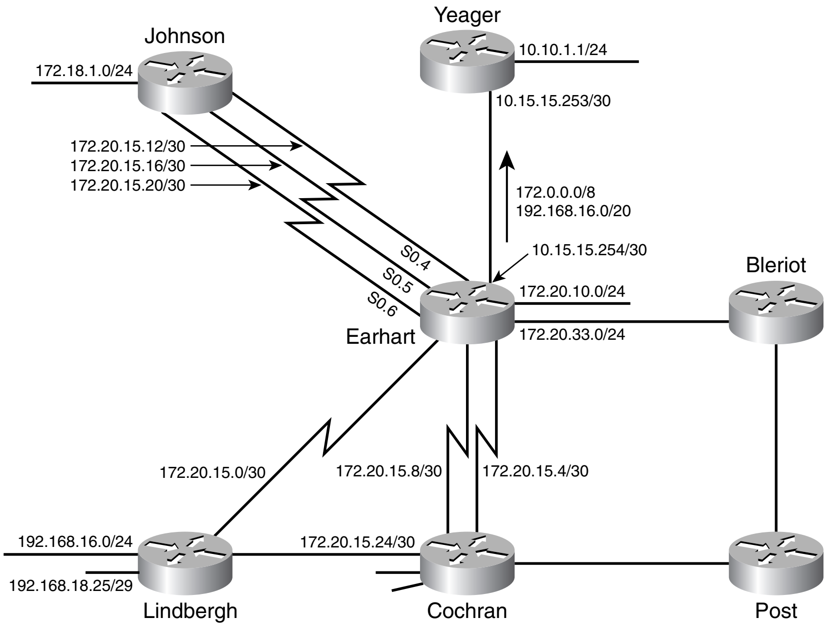
```
Earhart#show ip eigrp neighbor detail
IP-EIGRP neighbors for process 15
H   Address                 Interface   Hold Uptime   SRTT   RTO  Q  Seq Type
                                        (sec)         (ms)       Cnt Num
7   172.20.33.2             Et1           11 00:00:10   12   200  0  13
   Version 12.1/1.2, Retrans: 0, Retries: 0
6   10.15.15.253            Et2           11 00:00:10   12   200  0  6
   Version 12.3/1.2, Retrans: 1, Retries: 0
5   172.20.15.22            Se0.6         11 00:00:13  298  1788  0  73
   Version 12.3/1.2, Retrans: 0, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
4   172.20.15.14            Se0.4         11 00:00:13  927  5000  0  81
   Version 12.3/1.2, Retrans: 0, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
3   172.20.15.18            Se0.5         10 00:00:13  817  4902  0  80
   Version 12.3/1.2, Retrans: 0, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
0   172.20.15.2             Se0.2         11 00:15:48  274  1644  0  13
   Version 12.3/1.2, Retrans: 0, Retries: 0
2   172.20.15.10            Se0.3         13 00:45:40   72   570  0  59
   Version 12.3/1.2, Retrans: 0, Retries: 0
1   172.20.15.6             Se0.1         11 00:46:01   61   366  0  57
   Version 12.3/1.2, Retrans: 0, Retries: 0
Earhart#
```

Earhart has three stub neighbors. They are connected to the three links to Johnson: 172.20.15.22, 172.20.15.14, and 172.20.15.17.

Now, suppose a new link is added between Lindbergh and Cochran to add redundancy for traffic connected to Lindbergh's LAN traveling into the core of the network, as shown in Figure 7-37. Before Lindbergh is configured as a stub, when the links between Cochran and Earhart fail, Cochran will send queries to Lindbergh regarding alternate paths to the addresses in its topology table, 172.20.10.0 for instance. Lindbergh responds with a positive reply, because Lindbergh has an entry in its topology table for 172.20.10.0 via Earhart. Traffic from Cochran to 172.20.10.1 travels via Lindbergh.

Although this is an alternate path, forwarding traffic through a remote site is not always desirable. In this case, Lindbergh's links are there to allow redundancy from Lindbergh's addresses to core addresses, not to enable Lindbergh to act as a transit router. The bandwidth might not be sufficient to provide transit routing. Stub routing easily solves this problem. As a stub, no queries are sent to Lindbergh, so Lindbergh will not make itself available to Cochran as an alternate path. Furthermore, Lindbergh only sends updates containing connected, summary, static, or redistributed routes, not remote routes, such as 172.20.10.0.

**Figure 7-37** *A new link is added between Lindbergh and Cochran to add redundancy for traffic connected to Lindbergh's LAN traveling into the core of the network.*



Example 7-34 shows Cochran's EIGRP neighbors when Lindbergh (connected to Serial 0/0.4) is not configured as a stub router. Cochran's two links to Earhart are brought down. Cochran's subsequent topology table (Example 7-35) shows all addresses are accessible via Lindbergh (Serial 0/0.4).

**Example 7-34** *Cochran's EIGRP neighbor table shows its neighbors. Lindbergh is not a stub.*

```
Cochran#show ip eigrp neighbor detail
IP-EIGRP neighbors for process 15
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   172.20.15.26            Se0/0.4           10 00:00:18 1152  5000  0  34
   Version 12.3/1.2, Retrans: 0, Retries: 0
1   172.20.15.9             Se0/0.2           14 00:41:58  104   624  0  205
   Version 12.1/1.2, Retrans: 1, Retries: 0
0   172.20.15.5             Se0/0.1           11 00:49:12   99   594  0  206
   Version 12.1/1.2, Retrans: 2, Retries: 0
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   172.20.32.2             Fa0/1             13 00:42:01   60   360  0  14
   Version 12.1/1.2, Retrans: 2, Retries: 0
Cochran#
```

**Example 7-35** *After failure of the primary links between Cochran and Earhart, all addresses are accessible via the dual connected spoke router, Lindbergh.*

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.5 (Serial0/0.1) is down:
interface down
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.9 (Serial0/0.2) is down:
interface down
Cochran#show ip eigrp topology
IP-EIGRP Topology Table for AS(15)/ID(192.168.17.1)
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 10.0.0.0/8, 1 successors, FD is 2707456
        via 172.20.15.26 (2707456/2195456), Serial0/0.4
P 192.168.18.24/29, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.16.0/24, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.17.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/0
P 172.20.32.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/1
P 172.20.33.0/24, 1 successors, FD is 2707456
        via 172.20.15.26 (2707456/2195456), Serial0/0.4
P 172.20.15.20/30, 1 successors, FD is 21536000
        via 172.20.15.26 (21536000/21024000), Serial0/0.4
P 172.20.15.16/30, 1 successors, FD is 6535936
        via 172.20.15.26 (6535936/6023936), Serial0/0.4
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
P 172.20.10.0/27, 1 successors, FD is 2707456
        via 172.20.15.26 (2707456/2195456), Serial0/0.4
P 172.20.15.4/30, 1 successors, FD is 3193856
        via 172.20.15.26 (3193856/2681856), Serial0/0.4
P 172.20.15.0/30, 1 successors, FD is 2681856
        via 172.20.15.26 (2681856/2169856), Serial0/0.4
P 172.20.15.12/30, 1 successors, FD is 4035840
        via 172.20.15.26 (4035840/3523840), Serial0/0.4
P 172.18.0.0/16, 1 successors, FD is 4038400
        via 172.20.15.26 (4038400/3526400), Serial0/0.4
P 172.20.15.8/30, 1 successors, FD is 11535872
        via 172.20.15.26 (11535872/11023872), Serial0/0.4
P 192.168.18.128/25, 1 successors, FD is 28160
        via Connected, FastEthernet0/2
P 10.108.16.0/24, 1 successors, FD is 284160
        via 172.20.32.2 (284160/281600), FastEthernet0/1
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
Cochran#
```

Now, Lindbergh is configured as an EIGRP stub in Example 7-36.

**Example 7-36** *Lindbergh is configured as an EIGRP stub router.*

```
router eigrp 15
 eigrp stub
```

Example 7-37 shows Cochran's EIGRP neighbor table, and Example 7-38 shows Cochran's EIGRP topology table after the same two links to Earhart fail again.

**Example 7-37** *Cochran's EIGRP neighbor table shows its neighbors. Lindbergh is a stub. Cochran is running a later IOS release (12.3) then Earhart. Notice that the fact that queries are suppressed is explicitly stated.*

```
Cochran#show ip eigrp neighbor detail
IP-EIGRP neighbors for process 15
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
2   172.20.15.26            Se0/0.4           10 00:01:08   56   336  0  20
   Version 12.3/1.2, Retrans: 2, Retries: 0
   Stub Peer Advertising ( CONNECTED SUMMARY ) Routes
   Suppressing queries
1   172.20.15.9             Se0/0.2           11 00:29:46   96   576  0  111
   Version 12.1/1.2, Retrans: 1, Retries: 0
0   172.20.15.5             Se0/0.1           10 00:37:00   96   576  0  110
   Version 12.1/1.2, Retrans: 2, Retries: 0
IP-EIGRP neighbors for process 10
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q  Seq
                                            (sec)         (ms)       Cnt Num
0   172.20.32.2             Fa0/1             13 00:29:50   51   306  0  10
   Version 12.1/1.2, Retrans: 2, Retries: 0
Cochran#
```

**Example 7-38** *After failure of the primary links between Cochran and Earhart, only addresses connected to Lindbergh are reachable via Serial 0/0.4.*

```
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.5 (Serial0/0.1) is down:
interface down
%DUAL-5-NBRCHANGE: IP-EIGRP(0) 15: Neighbor 172.20.15.9 (Serial0/0.2) is down:
interface down

Cochran#show ip eigrp topology
IP-EIGRP Topology Table for AS(15)/ID(192.168.17.1)
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status

P 192.168.18.24/29, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.16.0/24, 1 successors, FD is 2195456
        via 172.20.15.26 (2195456/281600), Serial0/0.4
P 192.168.17.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/0
```

**Example 7-38** *After failure of the primary links between Cochran and Earhart, only addresses connected to Lindbergh are reachable via Serial 0/0.4. (Continued)*

```
P 172.20.32.0/24, 1 successors, FD is 28160
        via Connected, FastEthernet0/1
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
P 172.20.15.0/30, 1 successors, FD is 2681856
        via 172.20.15.26 (2681856/2169856), Serial0/0.4
P 192.168.18.128/25, 1 successors, FD is 28160
        via Connected, FastEthernet0/2
P 10.108.16.0/24, 1 successors, FD is 284160
        via 172.20.32.2 (284160/281600), FastEthernet0/1
P 172.20.15.24/30, 1 successors, FD is 2169856
        via Connected, Serial0/0.4
Cochran#
```

Configuring Lindbergh as an EIGRP stub prevents it from becoming a transit router during a failure of core links.

Configuring stub routing with EIGRP greatly increases the scalability of an EIGRP network, by minimizing queries, and thus the amount of time that network outages require addresses to be in an active state.

Stub routing eliminates queries sent to the stub router, but it does nothing to hide the topology of the rest of the network from the stub's point of view. Earhart can hide the topology of the rest of the network from the stubs. They don't need to know about every individual subnet because all packets for each of the subnets are always forwarded to the hub. Earhart can accomplish this by summarizing addresses.

## Case Study: Address Summarization

Router Yeager, shown in the network in Figure 7-37, has a single link to Earhart. The six addresses that Earhart must advertise to Yeager can be summarized with two aggregate addresses. Earhart's configuration will be as shown in Example 7-39.

**Example 7-39** *Earhart's configuration summarizes routes to Yeager.*

```
interface Ethernet2
 ip address 10.15.15.254 255.255.255.252
 ip summary-address eigrp 15 172.0.0.0 255.0.0.0
 ip summary-address eigrp 15 192.168.16.0 255.255.240.0
```

The **ip summary-address eigrp** command will automatically suppress the advertisement of the more specific networks to Yeager. Example 7-40 shows the route table of Yeager before and after the aggregate addresses are configured. Even in this small network, the number of EIGRP-learned entries has been reduced by half; in a large network, the impact on route tables and the memory required to store them can be significant.

**Example 7-40** *Yeager's route table before and after aggregate addresses are configured at Earhart.*

```
Yeager#show ip route
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, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

D    172.18.0.0/16 [90/3040000] via 10.15.15.254, 00:13:07, Ethernet0
D    172.20.0.0/16 [90/307200] via 10.15.15.254, 00:13:07, Ethernet0
     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.10.1.0/24 is directly connected, Ethernet1
C       10.15.15.252/30 is directly connected, Ethernet0
     192.168.17.0/27 is subnetted, 1 subnets
D       192.168.17.0 [90/2198016] via 10.15.15.254, 00:03:57, Ethernet0
D    192.168.16.0/24 [90/2221056] via 10.15.15.254, 00:01:51, Ethernet0
     192.168.18.0/24 is variably subnetted, 2 subnets, 2 masks
D       192.168.18.24/29 [90/2221056] via 10.15.15.254, 00:13:09, Ethernet0
D       192.168.18.128/25 [90/2198016] via 10.15.15.254, 00:13:09, Ethernet0

Yeager#show ip route
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, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       10.10.1.0/24 is directly connected, Ethernet1
C       10.15.15.252/30 is directly connected, Ethernet0
D    172.0.0.0/8 [90/307200] via 10.15.15.254, 00:00:57, Ethernet0
D    192.168.16.0/20 [90/2198016] via 10.15.15.254, 00:00:57, Ethernet0
```

## Authentication

MD5 cryptographic checksums are the only authentication supported in EIGRP, which on first consideration might seem less flexible than RIPv2 and OSPF, which support both MD5 and clear-text passwords. However, clear-text password authentication should be used only when a neighboring device does not support the more secure MD5. Because EIGRP will be spoken only between two Cisco devices, this situation will never arise.

The steps for configuring EIGRP authentication are

**Step 1** Define a key chain with a name.

**Step 2** Define the key or keys on the key chain.

**Step 3** Enable authentication on an interface and specify the key chain to be used.

**Step 4** Optionally configure key management.

Key-chain configuration and management are described in Chapter 6. EIGRP authentication is enabled and linked to a key chain on an interface with the commands **ip authentication key-chain eigrp** and **ip authentication mode eigrp md5**.[17]

Referring to Figure 7-37, the configuration in Example 7-41 enables EIGRP authentication on Cochran's interface to Earhart.

**Example 7-41** *Cochran is configured to use MD5 authentication with Earhart.*

```
key chain Edwards
 key 1
 key-string PanchoBarnes
!
interface Serial0/0.1
 ip address 172.20.15.6 255.255.255.252
 ip authentication key-chain eigrp 15 Edwards
 ip authentication mode eigrp 15 md5
interface Serial0/0.2
 ip address 172.20.15.10 255.255.255.252
 ip authentication key-chain eigrp 15 Edwards
 ip authentication mode eigrp 15 md5
```

A similar configuration would be necessary on Earhart. The commands **accept-lifetime** and **send-lifetime** are used for key-chain management as described in Chapter 6.

# Troubleshooting EIGRP

Troubleshooting the exchange of RIP route information is a reasonably simple procedure. Routing updates are either propagated or they are not, and they either contain accurate information or they do not. The added complexity of EIGRP means an added complexity to the troubleshooting procedure. Neighbor tables and adjacencies must be verified, the query/response procedure of DUAL must be followed, and the influences of VLSM on automatic summarization must be considered.

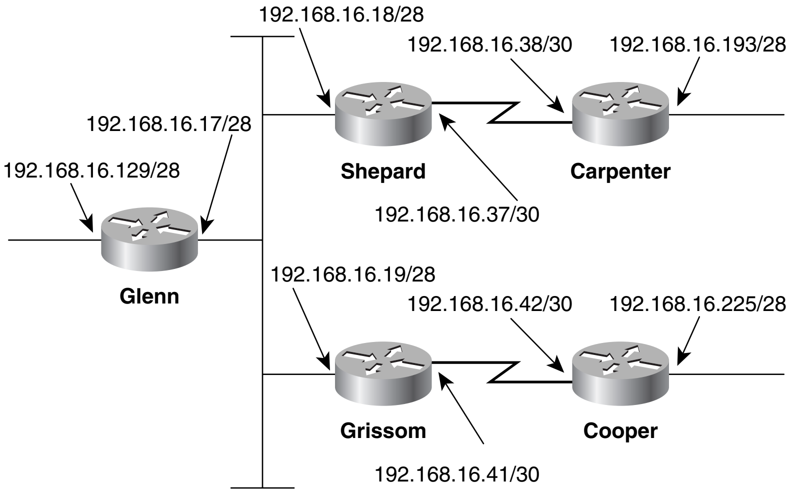
[17] Although MD5 is the only authentication mode available, the **ip authentication mode eigrp md5** command anticipates the possibility of another mode being available in the future.

This section's case study describes a sequence of events that typically can be used when pursuing an EIGRP problem. Following the case study is a discussion of an occasional cause of instabilities in larger EIGRP internets.

## Case Study: A Missing Neighbor

Figure 7-38 shows a small EIGRP network. Users are complaining that subnet 192.168.16.224/28 is unreachable. An examination of the route tables reveals that something is wrong at router Grissom (Example 7-42).[18]

**Figure 7-38** *Subnet 192.168.16.224/28 is not reachable through Grissom in this example of an EIGRP network.*



**Example 7-42** *The route tables of Shepard and Grissom show that Grissom's EIGRP process is not advertising or receiving routes on subnet 192.168.16.16/28.*

```
Grissom#show ip route
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
```

[18] When troubleshooting a network, it is a good practice to verify that the addresses of all router interfaces belong to the correct subnet.

**Example 7-42** *The route tables of Shepard and Grissom show that Grissom's EIGRP process is not advertising or receiving routes on subnet 192.168.16.16/28. (Continued)*

```
Gateway of last resort is not set
     192.168.16.0/24 is variably subnetted, 3 subnets, 2 masks
C       192.168.16.40/30 is directly connected, Serial0
C       192.168.16.16/28 is directly connected, Ethernet0
D       192.168.16.224/28 [90/2195456] via 192.168.16.42, 01:07:26, Serial0
```

```
Shepard#show ip route
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
Gateway of last resort is not set
     192.168.16.0/24 is variably subnetted, 4 subnets, 2 masks
C       192.168.16.36/30 is directly connected, Serial0
C       192.168.16.16/28 is directly connected, Ethernet0
D       192.168.16.192/28 [90/2297856] via 192.168.16.38, 01:07:20, Serial0
D       192.168.16.128/28 [90/307200] via 192.168.16.17, 01:07:20, Ethernet0
```

The following observations are made from the two route tables of Example 7-42:

- Shepard does not have subnets 192.168.16.40/30 and 192.168.16.224/28 in its route table, although Grissom does.
- Grissom's route table does not contain any of the subnets that should be advertised by Glenn or Shepard.
- Shepard's route table contains the subnets advertised by Glenn (and Glenn's table contains the subnets advertised by Shepard, although its route table is not included in Example 7-42).

The conclusion to be drawn from these observations is that Grissom is not advertising or receiving routes correctly over subnet 192.168.16.16/28.

Among the possible causes, the simplest causes should be examined first. These follow:

- An incorrect interface address or mask
- An incorrect EIGRP process ID
- A missing or incorrect network statement

In this case, there are no EIGRP or address configuration errors.

Next, the neighbor tables should be examined. Looking at the neighbor tables at Grissom, Shepard, and Glenn (Example 7-43), two facts stand out:

- Grissom (192.168.16.19) is in its neighbors' tables, but its neighbors are not in Grissom's neighbor table.

- The entire network has been up for more than five hours; this information is reflected in the *uptime* statistic for all neighbors except Grissom. However, Grissom's uptime shows approximately one minute.

**Example 7-43** *Shepard and Glenn see Grissom as a neighbor, but Grissom does not see them. This suggests that Shepard and Glenn are receiving Hellos from Grissom, but Grissom is not receiving Hellos from Shepard and Glenn.*

```
Grissom#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address           Interface    Hold   Uptime     SRTT    RTO    Q    Seq
                                   (sec)             (ms)          Cnt   Num
0   192.168.16.42     Se0            11   05:27:11    23     200    0    8
```

```
Shepard#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address           Interface    Hold   Uptime     SRTT    RTO    Q    Seq
                                   (sec)             (ms)          Cnt   Num
1   192.168.16.19     Et0            12   00:01:01     0    5000    1    0
2   192.168.16.17     Et0            11   05:27:33     8     200    0    6
0   192.168.16.38     Se0            14   05:27:34    22     200    0    10
```

```
Glenn#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address           Interface    Hold   Uptime     SRTT    RTO    Q    Seq
                                   (sec)             (ms)          Cnt   Num
1   192.168.16.19     Et0            14   00:00:59     0    8000    1    0
2   192.168.16.18     Et0            10   05:30:11     9      20    0    7
0   192.168.16.130   Et1           12   05:30:58    6      20   0    7
```

If Grissom is in Shepard's neighbor table, Shepard must be receiving Hellos from it. Grissom, however, is apparently not receiving Hellos from Shepard. Without this two-way exchange of Hello packets, an adjacency will not be established and route information will not be exchanged.

A closer examination of Shepard's and Glenn's neighbor tables reinforces this hypothesis:

- The SRTT for Grissom is 0, indicating that a packet has never made the round trip.
- The RTO for Grissom has increased to five and eight seconds, respectively.
- There is a packet enqueued for Grissom (Q Cnt).
- The sequence number recorded for Grissom is 0, indicating that no reliable packets have ever been received from it.

These factors indicate that the two routers are trying to send a packet reliably to Grissom, but are not receiving an ACK.

In Example 7-44, **debug eigrp packets** is used at Shepard to get a better look at what is happening. All EIGRP packet types will be displayed, but a second debug command is used with it: **debug ip eigrp neighbor 75 192.168.16.19**. This command adds a filter to the first command. It tells **debug eigrp packet** to display only IP packets of EIGRP 75 (the process

ID of the routers in Figure 7-38) and only those packets that concern neighbor 192.168.16.19 (Grissom).

**Example 7-44** *The command* **debug ip eigrp neighbor** *is used to control the packets displayed by debug eigrp packets.*

```
Shepard#debug eigrp packets
EIGRP Packets debugging is on
   (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK)
Shepard#debug ip eigrp neighbor 75 192.168.16.19
IP Neighbor target enabled on AS 75 for 192.168.16.19
IP-EIGRP Neighbor Target Events debugging is on
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19, retry 14, RTO 5000
   AS 75, Flags 0x1, Seq 22/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19, retry 15, RTO 5000
AS 75, Flags 0x1, Seq 22/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19, retry 16, RTO 5000
   AS 75, Flags 0x1, Seq 22/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/1
EIGRP: Retransmission retry limit exceeded
EIGRP: Received HELLO on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0
EIGRP: Enqueueing UPDATE on Ethernet0 nbr 192.168.16.19 iidbQ un/rely 0/1 peerQ
un/rely 0/0 serno 1-4
EIGRP: Sending UPDATE on Ethernet0 nbr 192.168.16.19
   AS 75, Flags 0x1, Seq 23/0 idbQ 1/0 iidbQ un/rely 0/0 peerQ un/rely 0/1 serno
1-4
```

Example 7-44 shows that Hello packets are being received from Grissom. It also shows that Shepard is attempting to send updates to Grissom; Grissom is not acknowledging them. After the 16th retry, the message "Retransmission retry limit exceeded" is displayed. This exceeded limit accounts for the low uptime shown for Grissom in the neighbor tables—when the retransmission retry limit is exceeded, Grissom is removed from the neighbor table. But because Hellos are still being received from Grissom, it quickly reappears in the table and the process begins again.

Example 7-45 shows the output from **debug eigrp neighbors** at Shepard. This command is not IP specific, but instead shows EIGRP neighbor events. Here, two instances of the events described in the previous paragraph are displayed: Grissom is declared dead as the retransmission limit is exceeded but is immediately "revived" when its next Hello is received.

**Example 7-45** **debug eigrp neighbors** *displays neighbor events.*

```
Shepard#debug eigrp neighbors
EIGRP Neighbors debugging is on
Shepard#
EIGRP: Retransmission retry limit exceeded
EIGRP: Holdtime expired
EIGRP: Neighbor 192.168.16.19 went down on Ethernet0
EIGRP: New peer 192.168.16.19
EIGRP: Retransmission retry limit exceeded
EIGRP: Holdtime expired
EIGRP: Neighbor 192.168.16.19 went down on Ethernet0
EIGRP: New peer 192.168.16.19
```

Although Example 7-44 shows that update packets are being sent to Grissom, observation of EIGRP packets at that router shows that they are not being received (Example 7-46).

**Example 7-46** *Grissom is exchanging Hellos with Cooper via interface S0 and is sending Hellos out E0. However, Grissom is not receiving any EIGRP packets on interface EO.*

```
Grissom#debug eigrp packets
EIGRP Packets debugging is on
     (UPDATE, REQUEST, QUERY, REPLY, HELLO, IPXSAP, PROBE, ACK)
Grissom#
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Received HELLO on Serial0 nbr 192.168.16.42
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Received HELLO on Serial0 nbr 192.168.16.42
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Received HELLO on Serial0 nbr 192.168.16.42
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0 peerQ un/rely 0/0
EIGRP: Sending HELLO on Serial0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
EIGRP: Sending HELLO on Ethernet0
   AS 75, Flags 0x0, Seq 0/0 idbQ 0/0 iidbQ un/rely 0/0
```

Because Grissom is successfully exchanging Hellos with Cooper, Grissom’s EIGRP process must be working. Suspicion therefore falls on Grissom’s Ethernet interface. An inspection of the configuration file shows that an access list is configured as an incoming filter on E0 in Example 7-47.

**Example 7-47** *An incoming access-list is denying EIGRP packets.*

```
interface Ethernet0
 ip address 192.168.16.19 255.255.255.240
 ip access-group 150 in
!
!
access-list 150 permit tcp any any established
access-list 150 permit tcp any host 192.168.16.238 eq ftp
access-list 150 permit tcp host 192.168.16.201 any eq telnet
access-list 150 permit tcp any host 192.168.16.230 eq pop3
access-list 150 permit udp any any eq snmp
access-list 150 permit icmp any 192.168.16.224 0.0.0.15
```

When EIGRP packets are received at Grissom's E0 interface, they are first filtered through access list 150. They will not match any entry on the list and are therefore being dropped. The problem is resolved (Example 7-48) by adding the following entry to the access list:

```
access-list 150 permit eigrp 192.168.16.16 0.0.0.15 any
```

**Example 7-48** *When an entry is added to the access list to permit EIGRP packets, Grissom's neighbor and route tables show that it now has routes to all subnets.*

```
Grissom#show ip eigrp neighbors
IP-EIGRP neighbors for process 75
H   Address                 Interface    Hold Uptime    SRTT    RTO    Q    Seq
                                         (sec)          (ms)          Cnt   Num
2   192.168.16.17           Et0             10 00:06:20    4    200    0    41
1   192.168.16.18           Et0             14 00:06:24   15    200    0    85
0   192.168.16.42           Se0             10 06:22:56   22    200    0    12
Grissom#show ip route
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
Gateway of last resort is not set
   192.168.16.0/24 is variably subnetted, 6 subnets, 2 masks
C    192.168.16.40/30 is directly connected, Serial0
D    192.168.16.36/30 [90/2195456] via 192.168.16.18, 00:06:27, Ethernet0
C    192.168.16.16/28 is directly connected, Ethernet0
D    192.168.16.224/28 [90/2195456] via 192.168.16.42, 00:06:12, Serial0
D    192.168.16.192/28 [90/2323456] via 192.168.16.18, 00:06:27, Ethernet0
D    192.168.16.128/28 [90/307200] via 192.168.16.17, 00:06:12, Ethernet0
Grissom#
```

## Stuck-in-Active Neighbors

When a route goes active and queries are sent to neighbors, the route will remain active until a reply is received for every query. But what happens if a neighbor is dead or otherwise incapacitated and cannot reply? The route would stay permanently active. The active timer and SIA-retransmit timer are designed to prevent this situation. Both the active timer and the SIA-retransmit timer are set when a query is sent. If the SIA-retransmit timer is not supported by the router's IOS (IOS versions earlier then 12.2[4.1]), only the active timer is used. If the timers expire before a reply to the query is received, the route is declared *stuck-in-active,* the neighbor is presumed dead, and it is flushed from the neighbor table.[19] The SIA route and any other routes via that neighbor are eliminated from the route table. DUAL will be satisfied by considering the neighbor to have replied with an infinite metric.

In reality, this sequence of events should never happen. The loss of Hellos should identify a disabled neighbor long before the active timer expires.

But what happens in large EIGRP networks where a query might, like the bunny in the battery advertisement, keep going and going? Remember that queries cause the diffusing calculation to grow larger, whereas replies cause it to grow smaller (refer to Figure 7-6). Queries must eventually reach the edge of the network, and replies must eventually begin coming back, but if the diameter of the diffusing calculation grows large enough, an active timer might expire before all replies are received. The result, flushing a legitimate neighbor from the neighbor table, is obviously destabilizing.

When neighbors mysteriously disappear from neighbor tables and then reappear, or users complain of intermittently unreachable destinations, SIA routes might be the culprit. Checking the error logs of routers is a good way to find out whether SIAs have occurred (Example 7-49).

**Example 7-49** *The final entry of this error log shows a SIA message.*

```
Gagarin#show logging
Syslog logging: enabled (0 messages dropped, 0 flushes, 0 overruns)
    Console logging: level debugging, 3369 messages logged
    Monitor logging: level debugging, 0 messages logged
    Trap logging: level informational, 71 message lines logged
    Buffer logging: level debugging, 3369 messages logged
Log Buffer (4096 bytes):
   ...
   ...
   ...
DUAL: dual_rcvupdate(): 10.51.1.0/24 via 10.1.2.1 metric 409600/128256
DUAL: Find FS for dest 10.51.1.0/24. FD is 4294967295, RD is 4294967295 found
DUAL: RT installed 10.51.1.0/24 via 10.1.2.1
DUAL: Send update about 10.51.1.0/24. Reason: metric chg
DUAL: Send update about 10.51.1.0/24. Reason: new if
DUAL: dual_rcvupdate(): 10.52.1.0/24 via 10.1.2.1 metric 409600/128256
DUAL: Find FS for dest 10.52.1.0/24. FD is 4294967295, RD is 4294967295 found
%DUAL-3-SIA: Route 10.11.1.0/24 stuck-in-active state in IP-EIGRP 1. Cleaning up
Gagarin#
```

[19] As mentioned previously, the default active time is three minutes. It can be changed with the command **timers active-time**.

When chasing the cause of SIAs, close attention should be paid to the topology table in routers. If routes can be "caught" in the active state, the neighbors from whom queries have not yet been received should be noted. For example, Example 7-50 shows a topology table in which several routes are active. Notice that most of them have been active for 15 seconds and that one (10.6.1.0) has been active for 41 seconds.

**Example 7-50** *This topology table shows several active routes, all of which are waiting for a reply from neighbor 10.1.2.1.*

```
Gagarin#show ip eigrp topology
IP-EIGRP Topology Table for process 1
Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply
       r - Reply Status
A 10.11.1.0/24, 0 successors, FD is 3072128000, Q
    1 replies, active 00:00:15, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.10.1.0/24, 0 successors, FD is 3584128000, Q
    1 replies, active 00:00:15, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.9.1.0/24, 0 successors, FD is 4096128000, Q
    1 replies, active 00:00:15, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.2.1.0/24, 1 successors, FD is Inaccessible, Q
    1 replies, active ve 00:00:15, query-origin: Local origin
    Remaining res:
        via 10.1.2.1, r, Ethernet0
P 10.1.2.0/24, 1 successors, FD is 281600
        via Connected, Ethernet0
A 10.6.1.0/24, 0 successors, FD is 3385160704, Q
    1 replies, active 00:00:41, query-origin: Local origin
    Remaining replies:
        via 10.1.2.1, r, Ethernet0
A 10.27.1.0/24, 0 successors, FD is 3897160704, Q
--More-
```

Notice also that in each case, the neighbor 10.1.2.1 has its reply status flag (r) set. That is the neighbor from which replies have not yet been received. There might be no problem with the neighbor itself or with the link to the neighbor, but this information points to the direction within the network topology in which the investigation should proceed.

Common causes of SIAs in larger EIGRP networks are heavily congested, low-bandwidth data links and routers with low memory or overutilized CPUs. The problem will be exacerbated if these limited resources must handle very large numbers of queries.

The careless adjustment of the bandwidth parameter on interfaces might be another cause of SIAs. Recall that EIGRP is designed to use no more than 50 percent of the available bandwidth of a link. This restriction means that EIGRP's pacing is keyed to the configured bandwidth. If the bandwidth is set artificially low in an attempt to manipulate routing choices, the EIGRP process might be starved. If IOS 11.2 or later is being run, the command **ip bandwidth-percent eigrp** may be used to adjust the percentage of bandwidth used.

For example, suppose that an interface is connected to a 56K serial link, but the bandwidth is set to 14K. EIGRP would limit itself to 50 percent of this amount, or 7K. The commands in Example 7-51 adjust the EIGRP bandwidth percent to 200 percent—200 percent of 14K, which is 50 percent of the actual bandwidth of the 56K link.

**Example 7-51** *Router configuration adjusts the percentage of the configured bandwidth that EIGRP will use.*

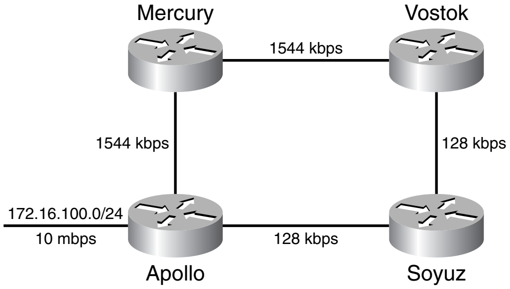
```
interface Serial 3
 ip address 172.18.107.210 255.255.255.240
 bandwidth 14
 ip bandwidth-percent eigrp 1 200
```

Increasing the active timer period with the **timers active-time** command might help avoid SIAs in some situations, but this step should not be taken without careful consideration of the effects it might have on reconvergence.

A new timer, the SIA-retransmit timer, and the two new EIGRP packet types, SIA-query and SIA-reply, help to minimize SIAs and to push the reset of the neighbor to link that is actually having the problem responding to queries.

Consider the network in Figure 7-39. From router Mercury, EIGRP will route traffic to network 172.16.100.0 via Apollo. Vostok is not a feasible successor because the metric from Vostok to 172.16.100.0 is too high. Vostok routes traffic to 172.16.100.0 via Mercury and Apollo. Soyuz is not a feasible successor because the metric from Soyuz to 172.16.100.0 is too high.

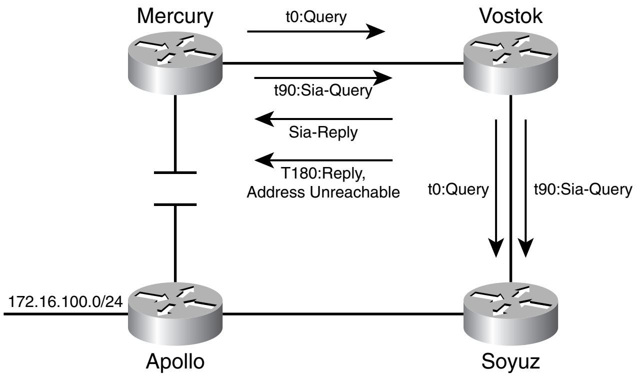
**Figure 7-39** *Mercury does not list Vostok as a feasible successor to 172.16.100.0/24.*



When the link between Mercury and Apollo fails, as shown in Figure 7-40, Mercury places the address 172.16.100.0 (and any other address known via neighbor Apollo) into Active, and sends a query to Vostok. Vostok also places the address into Active state, and sends a query to Soyuz. The Active timers are set. In addition, the SIA-retransmit timers are set. The SIA-retransmit timer is set to one-half the value of the Active timer, typically 90 seconds.

When the SIA-retransmit timer expires, Mercury sends an SIA-query to Vostok. Vostok sends an SIA-query to Soyuz. Vostok responds to the SIA-query from Mercury with an SIA-reply. Mercury resets the Active timer and the SIA-retransmit timer. The routers will send up to three SIA-queries (assuming no reply has been received from the original address query) as long as SIA-replies are received, before resetting a neighbor. So as long as a neighbor router responds to the SIA-queries, it won't be declared *stuck-in-active* and reset, for six minutes, assuming a default Active time of 180 seconds. This gives ample time for a large network to respond to queries.

**Figure 7-40** *SIA-queries and SIA-replies are used to avoid SIA conditions.*



But, say there is a problem on the link from Vostok to Soyuz that is allowing enough Hellos to get through to keep the neighbors active, but the SIA-reply is not received by Vostok within the SIA-retransmit time. If no SIA-reply is received within 90 seconds of a SIA-query, and no response to the original address query has been received, Vostok will reset neighbor Soyuz and reply to Mercury's original query that the address is unreachable.

The SIA-retransmit timer does two things. If neighbors are responding to SIA-queries, large networks are given more time to respond to address queries. If neighbors are not responding, the neighbor is reset. Only the router that is not receiving responses from its neighbor will reset the adjacency. Before the SIA-retransmit timer was introduced, any router that did not receive a response to an active query after the Active timer expired would reset the neighbor adjacency, even if the problem was somewhere downstream in the network.

A good network design is the best solution to instabilities such as SIA routes. By using a combination of intelligent address assignment, route filtering, default routes, stub routing, and summarization, boundaries may be constructed in a large EIGRP network to restrict the size and scope of diffusing computations. Chapter 13 includes an example of such a design.

## Looking Ahead

When comparing EIGRP and OSPF, it is often said that an advantage of EIGRP is that it is simpler to configure. This observation is true for many networks, but this chapter's discussion of troubleshooting shows that as a network grows, efforts must be made to "compartmentalize" the EIGRP topology. Ironically, the very complexity of OSPF might make it easier to configure in large networks, as the next chapter shows.

## Summary Table: Chapter 7 Command Review

| Command | Description |
|---|---|
| **accept-lifetime** *start-time* {**infinite** \| *end-time* \| **duration** *seconds*} | Specifies the time period during which the authentication key on a key chain is received as valid. |
| **auto-summary** | Enables automatic summarization at network boundaries. This command is enabled by default. |
| **bandwidth** *kilobits* | Specifies the bandwidth parameter, in kilobits per second, on an interface. |
| **debug eigrp packets** | Displays EIGRP packet activity. |
| **debug ip eigrp neighbor** *process-id address* | Adds a filter to the **debug eigrp packets** command, telling it to display only IP packets for the indicated process and neighbor. |
| **delay** *tens-of-microseconds* | Specifies the delay parameter, in tens of microseconds, on an interface. |
| **eigrp stub** {**connected** \| **redistributed** \| **static** \| **summary** \| **receive-only**} | Configures a spoke router as an EIGRP stub. |
| **ip authentication key-chain eigrp** *process-id key-chain* | Configures a key chain on an EIGRP interface and specifies the name of the key chain to be used. |
| **ip authentication mode eigrp** *process-id* ***md5*** | Enables EIGRP authentication on an interface. |
| **ip bandwidth-percent eigrp** *process-id percent* | Configures the percentage of bandwidth used by EIGRP; the default is 50 percent. |
| **ip hello-interval eigrp** *process-id seconds* | Configures the EIGRP hello interval. |
| **ip hold-time eigrp** *process-id seconds* | Configures the EIGRP hold time. |
| **ip summary-address eigrp** *process-id address mask* | Configures a router to send a summary EIGRP advertisement. |
| **key** *number* | Specifies a key on a key chain. |
| **key chain** *name-of-chain* | Specifies a group of authentication keys. |
| **key-string** *text* | Specifies the authentication string, or password, used by a key. |

*(Continued)*

| Command | Description |
|---|---|
| **metric weights** *tos k1 k2 k3 k4 k5* | Specifies how much weight the bandwidth, load, delay, reliability, and MTU size parameters should be given in the IGRP and EIGRP metric calculations. |
| **network** *network-number* | Specifies the network address of one or more interfaces on which IGRP, EIGRP, or RIP processes should be enabled. |
| **passive-interface** *type number* | Disables the transmission of broadcast or multicast routing updates on an interface. |
| **router eigrp** *process-id* | Enables an EIGRP process. |
| **send-lifetime** *start-time* {**infinite** \| *end-time* \| **duration** *seconds*} | Specifies the time period during which the authentication key on a key chain may be sent. |
| **show ip eigrp neighbors** [*type number*] | Displays the EIGRP neighbor table. |
| **show ip eigrp topology** [*process-id* \| [[*ip address*] *mask*]] | Displays the EIGRP topology table. |
| **timers active-time** {*minutes* \| **disabled**} | Changes or disables the default 3-minute active time. |
| **traffic-share {balanced \| min}** | Specifies whether an IGRP or EIGRP routing process should use unequal-cost load balancing or equal-cost load balancing only. |
| **variance** *multiplier* | Specifies a route multiplier by which a route metric can vary from the lowest-cost metric and still be included in an unequal-cost load balancing group. |

# Review Questions

1. Is EIGRP a distance vector or a link-state routing protocol?
2. What is the maximum configured bandwidth EIGRP will use on a link? Can this percentage be changed?
3. How do EIGRP and IGRP differ in the way they calculate the composite metric?
4. What are the four basic components of EIGRP?
5. In the context of EIGRP, what does the term *reliable delivery* mean? Which two methods ensure reliable delivery of EIGRP packets?
6. Which mechanism ensures that a router is accepting the most recent route entry?
7. What is the multicast IP address used by EIGRP?

**8** What are the packet types used by EIGRP?

**9** At what interval, by default, are EIGRP Hello packets sent?

**10** What is the default hold time?

**11** What is the difference between the neighbor table and the topology table?

**12** What is a feasible distance?

**13** What is the feasibility condition?

**14** What is a feasible successor?

**15** What is a successor?

**16** What is the difference between an active route and a passive route?

**17** What causes a passive route to become active?

**18** What causes an active route to become passive?

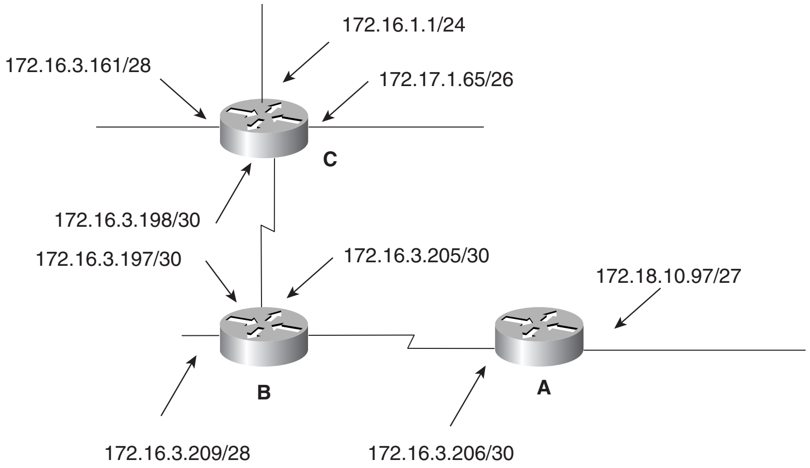
**19** What does stuck-in-active mean?

**20** What is the difference between subnetting and address aggregation?

# Configuration Exercises

**1** Write EIGRP configurations for Routers A, B, and C in Figure 7-41. Use process ID 5.

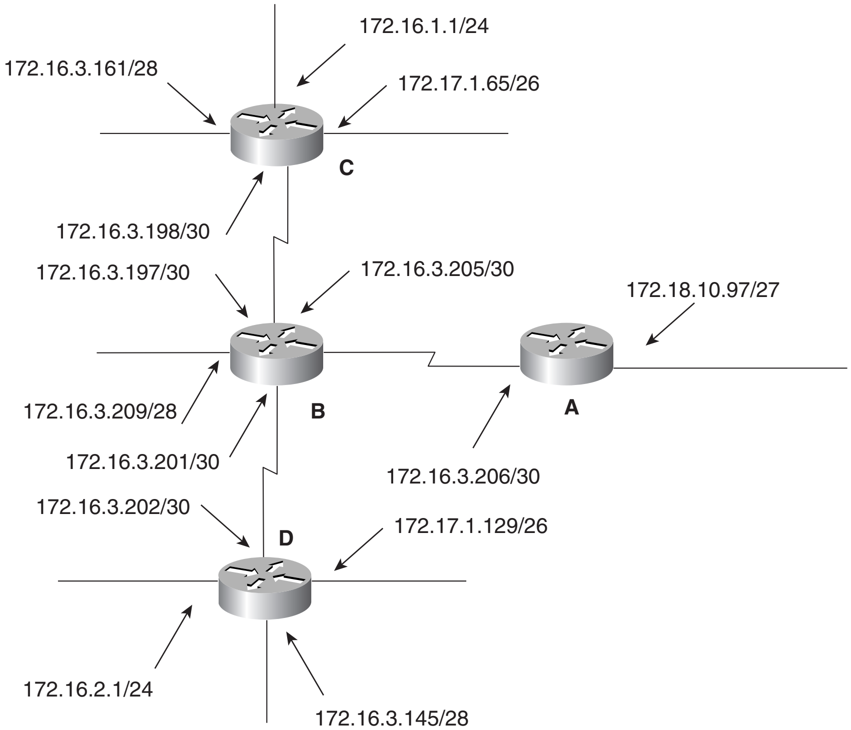
**Figure 7-41** *Network for Configuration Exercises 1 and 2.*



**2** The serial interfaces connecting Routers A and B in Figure 7-41 are both S0. Configure authentication between these two routers, using the first key two days from today's date. Configure a second key to be used beginning 30 days after the first key.

Router D is added in Figure 7-42. Add this router to the configurations written in Configuration Exercise 2.

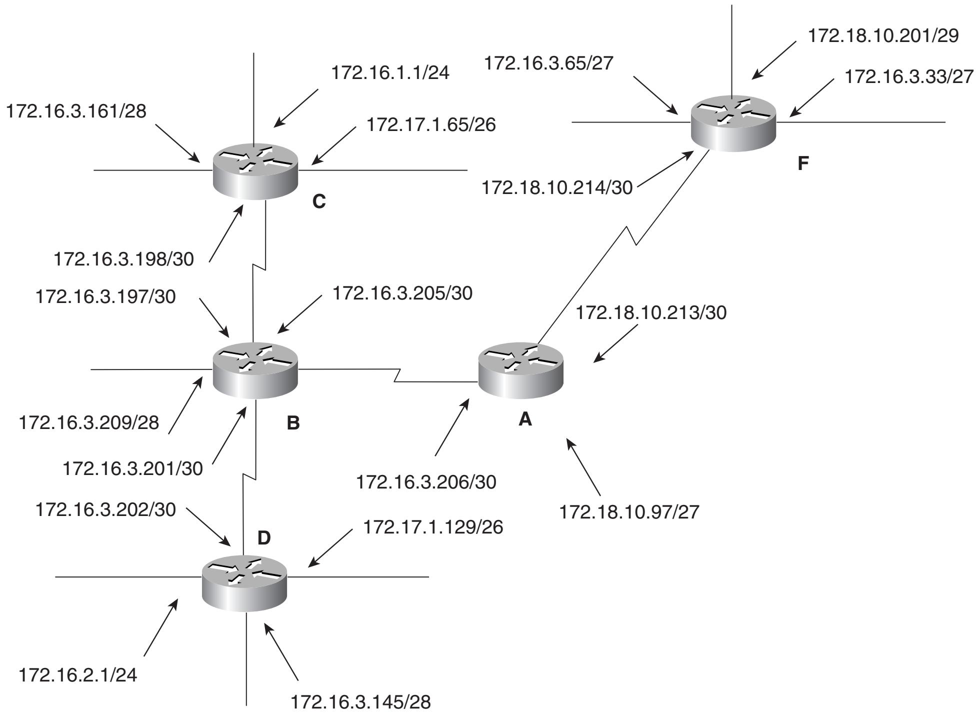
**Figure 7-42** *Network for Configuration Exercise 3.*



**3** Router F has been added in Figure 7-43. Configure this router to run EIGRP with the routers configured in Configuration Exercises 2 and 3.

**4** Configure route summarization wherever possible in the network of Figure 7-43.

**Figure 7-43** *Network for Configuration Exercises 4 and 5.*



# Troubleshooting Exercises

**1** Table 7-7 shows the values displayed in the **show interface** command for every interface in Figure 7-44. Which router will router F use as the successor to subnet A?

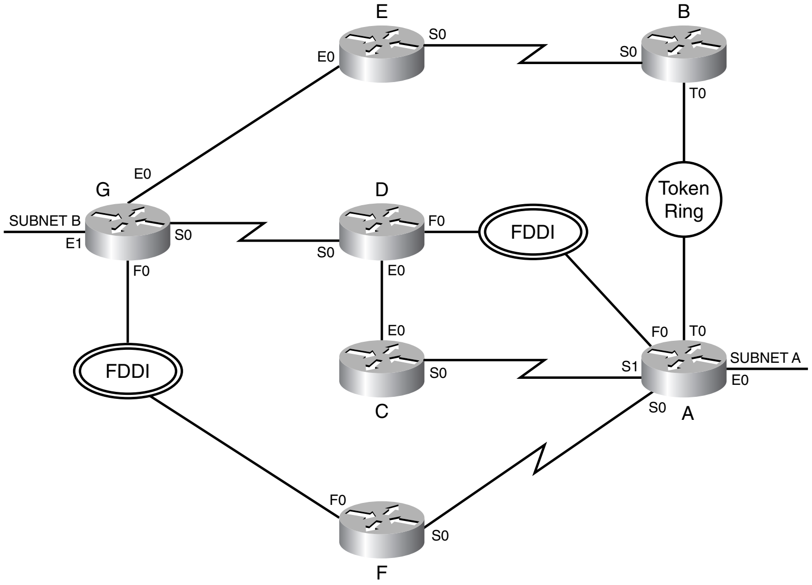
**Table 7-7** *Metric values for all interfaces in Figure 7-44, as displayed in the show interface command.*

| Router | Interface | BW(k) | DLY($\mu$S) |
|---|---|---|---|
| A | E0 | 10000 | 1000 |
| | F0 | 100000 | 100 |
| | T0 | 16000 | 630 |
| | S0 | 512 | 20000 |
| | S1 | 1544 | 20000 |
| B | TO | 16000 | 630 |
| | S0 | 1544 | 20000 |
| C | E0 | 10000 | 1000 |
| | S0 | 1544 | 20000 |

**Table 7-7** *Metric values for all interfaces in Figure 7-44, as displayed in the show interface command. (Continued)*

| Router | Interface | BW(k) | DLY($\mu$S) |
|---|---|---|---|
| D | E0 | 10000 | 1000 |
| | F0 | 100000 | 100 |
| | S0 | 1544 | 20000 |
| E | E0 | 10000 | 1000 |
| | S0 | 1544 | 20000 |
| F | F0 | 100000 | 100 |
| | S0 | 512 | 20000 |
| G | E0 | 10000 | 1000 |
| | E1 | 10000 | 1000 |
| | F0 | 100000 | 100 |
| | S0 | 56 | 20000 |

**Figure 7-44** *Network for Troubleshooting Exercises 1 through 5.*



**2** In Figure 7-44, what is router C's feasible distance to subnet A?

**3** In Figure 7-44, what is router G's feasible distance to subnet A?

**4** In Figure 7-44, which routers will router G show in its topology table as feasible successors?

**5** In Figure 7-44, what is router A's feasible distance to subnet B?