CISCO

Official Cert Guide

Learn, prepare, and practice for exam success

CCNP

Routing and Switching ROUTE 300-101

ciscopress.com

KEVIN WALLACE, CCIE® No. 7945

Contents

This chapter covers the following topics that you need to master for the CCNP ROUTE exam:

- **EIGRP Fundamentals:** This section reviews the EIGRP concepts, configuration, and verification commands covered in the CCNA curriculum.
- **EIGRP Neighborships:** This section discusses a variety of features that impact when a router attempts to form EIGRP neighbor relationships (neighborships), what must be true for those neighborships to work, and what might prevent those neighborships.
- **Neighborships over WANs:** This section examines the typical usage of EIGRP neighborships over various types of WAN technologies.

CHAPTER 4

Fundamental EIGRP Concepts

Enhanced Interior Gateway Routing Protocol (EIGRP) is configured with a few relatively simple commands. In fact, for most any size network, you could go to every router, enter the **router eigrp 1** command, followed by one or more **network** *net-id* subcommands (one for each classful network to which the router is connected), and EIGRP would likely work, and work very well, with no other configuration.

In spite of that apparent simplicity, here you sit beginning the first of four chapters of EIGRP coverage in this book. Many reasons exist for the amount of EIGRP material included here. First, EIGRP includes many optional configuration features that you need to both understand and master for the CCNP ROUTE exam. Many of these features require a solid understanding of EIGRP internals as well—a topic that can be conveniently ignored if you just do the minimal configuration, but something very important to planning, implementing, and optimizing a medium/large enterprise network.

Another reason for the depth of EIGRP coverage in this book is a fundamental change in the philosophy of the CCNP exams, as compared with earlier CCNP exam versions. Cisco has increased the focus on planning for the implementation and verification of new network designs. The bar has been raised, and in a way that is consistent with typical engineering jobs. Not only do you need to understand all the EIGRP features, but you also need to be able to look at a set of design requirements, and from that decide which EIGRP configuration settings could be useful—and which are not useful. You must also be able to direct others as to what verification steps would tell them if the implementation worked or not, rather than just relying on typing a **?** and looking around for that little piece of information you know exists somewhere.

This chapter begins with the "EIGRP Fundamentals" section, which is a review of the core prerequisite facts about EIGRP. Following the review, the chapter examines EIGRP neighbor relationships, including a variety of configuration commands that impact neighbor relationships, and the verification commands that you can use to confirm how well EIGRP neighbors work.

"Do I Know This Already?" Quiz

The "Do I Know This Already?" quiz enables you to assess whether you should read the entire chapter. If you miss no more than one of these seven self-assessment questions, you might want to move ahead to the "Exam Preparation Tasks" section. Table 4-1 lists the major headings in this chapter and the "Do I Know This Already?" quiz questions covering the material in those headings so that you can assess your knowledge of

these specific areas. The answers to the "Do I Know This Already?" quiz appear in Appendix A.

Table 4-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions
EIGRP Fundamentals	1, 2
EIGRP Neighborships	3–6
Neighborships over WANs	7

1. A router has been configured with the commands **router eigrp 9** and **network 172.16.1.0 0.0.0.255**. No other EIGRP-related commands have been configured. The answers list the IP addresses that could be assigned to this router's Fa0/0 interface. Which answers list an IP address/prefix length that would cause the router to enable EIGRP on Fa0/0? (Choose two answers.)

   a. 172.16.0.1/23

   b. 172.16.1.1/26

   c. 172.16.1.1/24

   d. 172.16.0.255/23

   e. None of the other answers are correct.

2. Router R1 has working interfaces S0/0, S0/1, and S0/2, with IP address/prefix combinations of 10.10.10.1/24, 10.10.11.2/24, and 10.10.12.3/22. R1's configuration includes the commands **router eigrp 9** and **network 10.0.0.0**. The **show ip eigrp interfaces** command lists S0/0 and S0/1 in the command output, but not S0/2. Which answer gives a possible reason for the omission?

   a. R1 has EIGRP neighbors reachable through S0/0 and S0/1, but not through S0/2, so it is not included.

   b. S0/2 might currently be in a state other than up/up.

   c. The **network 10.0.0.0** command requires the use of mask 255.0.0.0 because of EIGRP being classful by default.

   d. S0/2 might be configured as a passive interface.

3. Routers R1 and R2 are EIGRP neighbors using their Fa0/0 interfaces, respectively. An engineer adds the **ip hello-interval eigrp 9 6** command to R1's Fa0/0 configuration. Which of the following is true regarding the results from this change?

   a. The **show ip eigrp neighbors** command on R1 lists the revised Hello timer.

   b. The **show ip eigrp interfaces** command on R1 lists the revised Hello timer.

   c. The R1-R2 neighborship fails because of a Hello timer mismatch.

   d. The **show ip eigrp interfaces detail** command on R1 lists the revised Hello timer.

4. Router R1 has been configured with the commands **router eigrp 9** and network **172.16.2.0 0.0.0.255**, with no other current EIGRP configuration. R1's (working) Fa0/0 interface has been configured with IP address 172.16.2.2/26. R1 has found three EIGRP neighbors reachable through interface Fa0/0, including the router with IP address 172.16.2.20. When the engineer attempts to add the **neighbor 172.16.2.20 fa0/0** command in EIGRP configuration mode, which of the following occurs?

   a. Fa0/0 fails.

   b. The command is rejected.

   c. The existing three neighbors fail.

   d. The neighborship with 172.16.2.20 fails and then reestablishes.

   e. None of the other answers is correct.

5. Which of the following settings could prevent two potential EIGRP neighbors from becoming neighbors? (Choose two answers.)

   a. The interface used by one router to connect to the other router is passive in the EIGRP process.

   b. Duplicate EIGRP router IDs.

   c. Mismatched Hold Timers.

   d. IP addresses of 10.1.1.1/24 and 10.2.2.2/24, respectively.

6. An engineer has added the following configuration snippet to an implementation planning document. The configuration will be added to Router R1, whose Fa0/0 interface connects to a LAN to which Routers R2 and R3 also connect. R2 and R3 are already EIGRP neighbors with each other. Assuming that the snippet shows all commands on R1 related to EIGRP authentication, which answer lists an appropriate comment to be made during the implementation plan peer review?

```
key chain fred
key 3
key-string whehew
interface fa0/0
ip authentication key-chain eigrp 9 fred
```

a. The configuration is missing one authentication-related configuration command.

b. The configuration is missing two authentication-related configuration commands.

c. Authentication type 9 is not supported; type 5 should be used instead.

d. The key numbers must begin with key 1, so change the key 3 command to key 1.

7. A company has a Frame Relay WAN with one central-site router and 100 branch office routers. A partial mesh of PVCs exists: one PVC between the central site and each of the 100 branch routers. Which of the following could be true about the number of EIGRP neighborships?

a. A partial mesh totaling 100: one between the central-site router and each of the 100 branches.

b. A full mesh — (101 * 100) / 2 = 5050 — One neighborship between each pair of routers.

c. 101 — One between each router (including the central site) and its nearby PE router.

d. None of the answers is correct.

Foundation Topics

EIGRP Fundamentals

All the CCNP exams consider CCNA materials as prerequisites. So this book also assumes that the reader is already familiar with CCNA topics. However, the CCNP exams do test on features that overlap with CCNA. Additionally, most people forget some details along the way. Therefore, this section reviews the CCNA-level topics as a brief refresher.

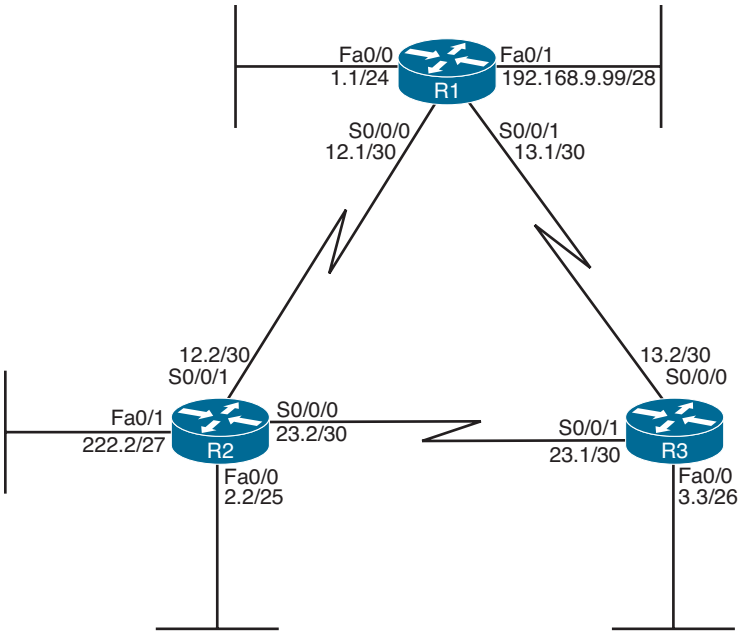
To that end, this section begins with a review of EIGRP configuration using only the **router eigrp** and **network** commands. Following that, the next section details the key fields used to verify that EIGRP is working. Finally, the last part of this introduction summarizes the basic EIGRP internals behind this initial simple example.

Configuration Review

Cisco IOS uses the **router eigrp** *asn* command (where *asn* is an autonomous system number [ASN]), plus one or more **network** *net-id wildcard-mask* subcommands, to enable EIGRP on the router and on router interfaces. The rules for these commands are as follows:

1. Neighboring routers' **router eigrp** *asn* commands must be configured with the same ASN parameter to become neighbors.
2. Cisco IOS enables only EIGRP on interfaces matched by an EIGRP **network** command. When enabled, the router does the following:
   a. Attempts to discover EIGRP neighbors on that interface by sending multicast EIGRP Hello messages
   b. Advertises to other neighbors about the subnet connected to the interface
3. If no wildcard mask is configured on the EIGRP **network** command, the command's single parameter should be a classful network number (in other words, a class A, B, or C network number).
4. If no wildcard mask is configured on the EIGRP **network** command, the command enables EIGRP on all of that router's interfaces directly connected to the configured classful network.
5. If the **network** command includes a wildcard mask, the router performs access control list (ACL) logic when comparing the *net-id* configured in the **network** command with each interface's IP address, using the configured wildcard mask as an ACL wildcard mask.

Example 4-1 shows a sample configuration for each router in Figure 4-1, with several variations in the **network** commands to make the details in the preceding list more obvious.



Note: All IP addresses begin with 10.1 unless otherwise noted.

Figure 4-1 *Three-Router Internetwork*

Example 4-1 *EIGRP Configuration on Routers R1, R2, and R3*

```
! On Router R1: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router eigrp 1
 network 10.0.0.0
 network 192.168.9.0
```

```
! On Router R2: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router eigrp 1
 network 10.1.0.0 0.0.31.255
 network 10.1.2.2 0.0.0.0
```

```
! On Router R3: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router eigrp 1
 network 10.1.0.0 0.0.255.255
```

First, note that all three routers use the **router eigrp 1** command, so all three routers' ASN values match.

Next, consider the two **network** commands on R1. The **network 10.0.0.0** command, without a *wildcard-mask* parameter, means that R1 matches all interfaces in class A network 10.0.0.0—which in this case means R1's Fa0/0, S0/0/0, and S0/0/1 interfaces. The **network 192.168.9.0** command, again without a wildcard mask, matches interface Fa0/1.

On R2, the **network 10.1.0.0 0.0.31.255** command requires a little more thought. The router uses the 0.0.31.255 value—the wildcard (WC) mask—just like an ACL WC mask. Cisco IOS compares the 10.1.0.0 value with each interface IP address, but only for the bit positions for which the WC mask lists a binary 0. For example, 0.0.31.255 represents 19 binary 0s, followed by 13 binary 1s. So, R2 would compare the first 19 bits of 10.1.0.0 with the first 19 bits of each interface's IP address.

Two features of the mechanics of the **network** command require a little extra attention. First, Cisco IOS might convert the address portion of the **network** *address wc-mask* command before putting the command into the running config. Just as Cisco IOS does for the address/WC mask combinations for the **access-list** command, Cisco IOS inverts the WC mask and then performs a Boolean AND of the address and mask. For example, if you type the **network 10.1.1.1 0.0.255.255** command, Cisco IOS inverts the WC mask (to 255.255.0.0) and ANDs this value with 10.1.1.1, resulting in 10.1.0.0. As a result, Cisco IOS stores the command **network 10.1.0.0 0.0.255.255**.

The second feature is that when you know for sure the values in the **network** command, you can easily find the range of interface addresses that match the address/WC mask combination in the **network** command. The low end of the range is the address as listed in the **network** command. To find the high end of the range, just add the address and WC mask together. For example, the **network 10.1.0.0 0.0.31.255** command has a range of 10.1.0.0 through 10.1.31.255.

Finally, on R3, the **network 10.1.0.0 0.0.255.255** command tells R3 to enable EIGRP on all interfaces whose IP addresses begin with 10.1, which includes all three interfaces on R3, as shown in Figure 4-1.

Taking a step back from the details, this config has enabled EIGRP, with ASN 1, on all three routers, and on all interfaces shown in Figure 4-1—except one interface. R2's Fa0/1 interface is not matched by any **network** commands on R2. So, EIGRP is not enabled on that interface. The next section reviews the commands that can be used to confirm that EIGRP is enabled, the interfaces on which it is enabled, the neighbor relationships that have been formed, and which EIGRP routes have been advertised and learned.

Verification Review

Even before starting to configure the routers, an engineer first considers all requirements. Those requirements lead to a design, which in turn leads to a chosen set of configuration commands. Then, the verification process that follows must consider the design requirements. The goal of verification is to determine that the internetwork works as designed, not just that some EIGRP routes have been learned.

For the purposes of this section, assume that the only design goal for the internetwork shown in Figure 4-1 is that EIGRP be used so that all routers have routes to reach all subnets shown in the figure.

To verify such a simple design, an engineer should start by confirming on which interfaces EIGRP has been enabled on each router. The next step should be to determine

whether the EIGRP neighbor relationships that should occur are indeed up and working. Then, the EIGRP topology table should be examined to confirm that there is at least one entry for each subnet or network in the design. Finally, the IP routes on each router should be examined, confirming that all routes are known. To that end, Table 4-2 summarizes five key **show** commands that provide the information to answer these questions.

Note The following table mentions some information that is covered later in this chapter (passive interfaces) or in other chapters (successor/feasible successors).

Example 4-2 shows samples of each command listed in Table 4-2. Note that the output highlights various samples of items that should be verified: the interfaces on which EIGRP is enabled, the known neighbors, the subnets in the topology table, and the EIGRP routes.

Key Topic

Table 4-2 *Key EIGRP Verification Commands*

Command	Key Information
show ip eigrp interfaces	Lists the working interfaces on which EIGRP is enabled (based on the **network** commands); it omits passive interfaces.
show ip protocols	Lists the contents of the **network** configuration commands for each routing process, and a list of neighbor IP addresses.
show ip eigrp neighbors	Lists known neighbors; does not list neighbors for which some mismatched parameter is preventing a valid EIGRP neighbor relationship.
show ip eigrp topology	Lists all successor and feasible successor routes known to this router. It does not list all known topology details. (See Chapter 5, "Advanced EIGRP Concepts," for more detail on successors and feasible successors.)
show ip route	Lists the contents of the IP routing table, listing EIGRP-learned routes with a code of D on the left side of the output.

Example 4-2 *EIGRP Verification on Routers R1, R2, and R3*

```
! On Router R1: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R1# show ip eigrp interfaces
IP-EIGRP interfaces for process 1
```

```
                    Xmit Queue      Mean    Pacing Time    Multicast    Pending
Interface     Peers   Un/Reliable     SRTT    Un/Reliable    Flow Timer    Routes
Fa0/0           0       0/0             0       0/1              0            0
Se0/0/0         1       0/0            25       0/15           123           0
Se0/0/1         1       0/0            23       0/15           111           0
Fa0/1   0         0/0       0         0/1     0        0
```

```
! On Router R2: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R2# show ip protocols
Routing Protocol is "eigrp 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 1
  EIGRP NSF-aware route hold timer is 240s
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    10.1.2.2/32
    10.1.0.0/19
  Routing Information Sources:
    Gateway         Distance      Last Update
    10.1.12.1              90      00:19:36
    10.1.23.1              90      00:19:36
  Distance: internal 90 external 170
```

```
! On Router R3: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R3# show ip eigrp neighbors
IP-EIGRP neighbors for process 1
H   Address                 Interface       Hold Uptime   SRTT   RTO  Q    Seq
                                            (sec)         (ms)       Cnt  Num
1   10.1.23.2               Se0/0/1         11 00:19:53    31    200   0    6
0   10.1.13.1               Se0/0/0         10 00:19:53    32    200   0    6
```

```
! On Router R2: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R2# show ip eigrp topology
IP-EIGRP Topology Table for AS(1)/ID(10.1.222.2)

Codes: P - Passive, A - Active, U - Update, Q - Query, R - Reply,
       r - reply Status, s - sia Status
```

```
P 10.1.13.0/30, 2 successors, FD is 2681856
        via 10.1.23.1 (2681856/2169856), Serial0/0/0
        via 10.1.12.1 (2681856/2169856), Serial0/0/1
P 10.1.12.0/30, 1 successors, FD is 2169856
        via Connected, Serial0/0/1
P 10.1.3.0/26, 1 successors, FD is 2172416
        via 10.1.23.1 (2172416/28160), Serial0/0/0
P 10.1.2.0/25, 1 successors, FD is 28160
        via Connected, FastEthernet0/0
P 10.1.1.0/24, 1 successors, FD is 2172416
        via 10.1.12.1 (2172416/28160), Serial0/0/1
P 10.1.23.0/30, 1 successors, FD is 2169856
        via Connected, Serial0/0/0
P 192.168.9.0/24, 1 successors, FD is 2172416
        via 10.1.12.1 (2172416/28160), Serial0/0/1
```

```
! On Router R3: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R3# show ip route
Codes: C - connected, S - static, 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
       i - IS-IS, su - IS-IS summary, 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    192.168.9.0/24 [90/2172416] via 10.1.13.1, 00:19:55, Serial0/0/0
     10.0.0.0/8 is variably subnetted, 6 subnets, 4 masks
C       10.1.13.0/30 is directly connected, Serial0/0/0
D       10.1.12.0/30 [90/2681856] via 10.1.23.2, 00:19:55, Serial0/0/1
                     [90/2681856] via 10.1.13.1, 00:19:55, Serial0/0/0
C       10.1.3.0/26 is directly connected, FastEthernet0/0
D       10.1.2.0/25 [90/2172416] via 10.1.23.2, 00:19:55, Serial0/0/1
D       10.1.1.0/24 [90/2172416] via 10.1.13.1, 00:19:55, Serial0/0/0
C       10.1.23.0/30 is directly connected, Serial0/0/1
```

To verify the interfaces on which EIGRP is enabled, both the **show ip eigrp interfaces** command (shown on R1) and the **show ip protocols** command (shown on R2) list the information. For this example, look at the list of interfaces in R2's **show ip protocols** command output: S0/0/0, S0/0/1, and FA0/0 are listed, but Fa0/1—unmatched by any of R2's network commands—is not.

In this design, each router should form a neighbor relationship with the other two routers, in each case over a point-to-point serial link. The **show ip eigrp neighbors** command (on R3) confirms R3's neighbors.

Finally, one design goal was for all routers to have routes for all subnets/networks. You could move on to the **show ip route** command or first look for all prefixes in the **show ip eigrp topology** command. With relatively general requirements, just looking at the IP routing table is fine. The example highlights R3's topology data and IP route for subnet 10.1.1.0/24. Of more interest might be the fact that the **show ip route** command output on R3 lists all subnet/network numbers except one: subnet 10.1.222.0/27. This subnet exists off R2's Fa0/1 interface (as seen in Figure 4-1), which is the interface on which EIGRP has not yet been enabled.

Internals Review

To complete the review of prerequisite CCNA-level EIGRP knowledge, this section looks at a few of the internals of EIGRP. Some of the facts listed here simply need to be memorized, whereas other topics will be discussed in more detail later.

EIGRP follows three general steps to add routes to the IP routing table, as follows:

Step 1. **Neighbor discovery:** EIGRP routers send Hello messages to discover potential neighboring EIGRP routers and perform basic parameter checks to determine which routers should become neighbors.

Step 2. **Topology exchange:** Neighbors exchange full topology updates when the neighbor relationship comes up, and then only partial updates as needed based on changes to the network topology.

Step 3. **Choosing routes:** Each router analyzes its respective EIGRP topology table, choosing the lowest-metric route to reach each subnet.

Because the majority of the rest of this chapter examines EIGRP neighborships, this review section skips any discussion of EIGRP neighbors, instead focusing on topology exchange and route selection.

Exchanging Topology Information

First, the EIGRP neighbor table lists the neighboring routers. Second, the EIGRP topology table holds all the topology information learned from EIGRP neighbors. Finally, EIGRP chooses the best IP routes, and those routes become candidates to be injected into the IP routing table. (Table 4-2, earlier in this chapter, lists the **show** commands that can be used to examine these tables.) EIGRP routers follow the process shown in Figure 4-2 to build the necessary information in these tables, with the end goal of populating the IP routing table.

EIGRP uses Update messages to send topology information to neighbors. These Update messages can be sent to multicast IP address 224.0.0.10 if the sending router needs to update multiple routers on the same subnet. Unlike OSPF, there is no concept of a designated router (DR) or backup designated router (BDR), but the use of multicast packets on LANs allows EIGRP to exchange routing information with all neighbors on the LAN efficiently.

The update messages are sent using the *Reliable Transport Protocol (RTP)*. The significance of RTP is that, like OSPF, EIGRP resends routing updates that are lost in transit.

By using RTP to guarantee delivery of the EIGRP messages, EIGRP can better avoid loops.

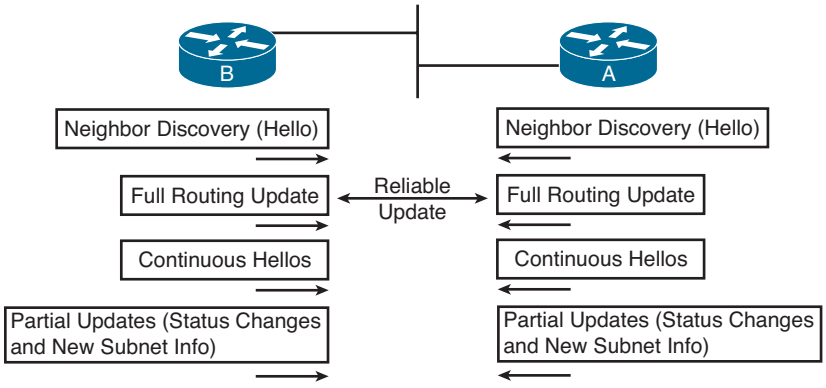


Figure 4-2 *EIGRP Discovery and Update Process*

Note The acronym RTP also refers to a different protocol, Real-time Transport Protocol (RTP), which is used to transmit voice and video IP packets.

Neighbors use both full routing updates and partial updates, as depicted in Figure 4-2. A full update means that a router sends information about all known routes, whereas a partial update includes only information about recently changed routes. Full updates occur when neighbors first come up. After that, the neighbors send only partial updates in reaction to changes to a route.

Calculating the Best Routes for the Routing Table

EIGRP topology information includes the subnet number and mask, along with the components of the EIGRP composite metric. Each router then calculates an integer metric for each route, using the individual values of the EIGRP metric components listed in the EIGRP topology database. By default, EIGRP only uses the bandwidth and delay settings when calculating the metric. Optionally, the calculation can also include interface load and interface reliability, although Cisco recommends against using either.

Note Past documents and books often stated that EIGRP, and its predecessor IGRP, also could use Maximum Transmission Unit (MTU) as a part of the metric. However, MTU size is intended to be a tiebreaker if two paths have equal metrics but different MTU sizes. In such a case, the path with the higher MTU is selected. So, while MTU size is listed in EIGRP Update messages, it is not directly used in metric calculations.

EIGRP calculates the metric for each possible route by inserting the values of the composite metric into a formula. If the choice is made to just use the default parameters of bandwidth and delay, the formula is as follows:

$$\text{Metric} = \left(\left(\frac{10^7}{\text{least-bandwidth}} \right) + \text{cumulative-delay} \right) * 256$$

In this formula, the term *least-bandwidth* represents the lowest-bandwidth link in the route, using a unit of kilobits per second. For example, if the slowest link in a route is a 10-Mbps Ethernet link, the first part of the formula is 10^7 / 10^4, because 10 Mbps equals 10,000 kbps, or 10^4 kbps. The *cumulative-delay* value used by the formula is the sum of all the delay values for all links in the route, with a unit of "tens of microseconds." So, if you add up all the delays (from the output of the **show interfaces** *type number* command) from all egress interfaces, you would take that number (which is in microseconds) and divide by 10 (to give you a unit of *tens of microseconds*) for use in the formula. You can set both bandwidth and delay for each link, using the **bandwidth** and **delay** interface subcommands.

Table 4-3 summarizes some of the key facts about EIGRP.

Table 4-3 *EIGRP Feature Summary*

Feature	Description
Transport	IP, protocol type 88 (does not use UDP or TCP).
Metric	Based on constrained bandwidth and cumulative delay by default, and optionally load and reliability.
Hello interval	Interval at which a router sends EIGRP Hello messages on an interface.
Hold Timer	Timer used to determine when a neighboring router has failed, based on a router not receiving any EIGRP messages, including Hellos, in this timer period.
Update destination address	Normally sent to 224.0.0.10, with retransmissions being sent to each neighbor's unicast IP address. Can also be sent to the neighbor's unicast IP address.
Full or partial updates	Full updates are used when new neighbors are discovered; otherwise, partial updates are used.
Authentication	Supports MD5 authentication only.
VLSM/classless	EIGRP includes the mask with each route, also allowing it to support discontiguous networks and VLSM.
Route tags	Allows EIGRP to tag routes as they are redistributed into EIGRP.

Feature	Description
Next-hop field	Supports the advertisement of routes with a different next-hop router than the advertising router.
Manual route summarization	Allows route summarization at any point in the EIGRP network.
Automatic summarization	EIGRP supports, and defaults to use, automatic route summarization at classful network boundaries.
Multiprotocol	Supports the advertisement of IPX, AppleTalk, IP version 4, and IP version 6 routes.

This completes the CCNA-level EIGRP review. The rest of this chapter now examines EIGRP neighbor relationships.

EIGRP Neighborships

Like OSPF, EIGRP uses three major steps to achieve its goal of learning the best available loop-free routes:

Step 1. Establish EIGRP neighbor relationships—*neighborships*—with other routers that share a common subnet.

Step 2. Exchange EIGRP topology data with those neighbors.

Step 3. Calculate the currently best IP route for each subnet, based on the known EIGRP topology data, and add those best routes to the IP routing table.

This three-step process hinges on the first step—the successful creation of neighbor relationships between EIGRP routers. The basic EIGRP configuration described earlier in this chapter, particularly the **network** command, most directly tells EIGRP on which interfaces to dynamically discover neighbors. After EIGRP neighborships have been formed with neighboring routers that are reachable through those interfaces, the final two steps occur without any additional direct configuration.

EIGRP dynamically discovers neighbors by sending EIGRP Hello messages on each EIGRP-enabled interface. When two routers hear EIGRP Hello messages from each other, they check the EIGRP parameters listed in those messages and decide whether the two routers should or should not become neighbors.

The rest of this section focuses on topics related to EIGRP neighborship, specifically:

- Manipulating EIGRP Hello and Hold Timers
- Controlling whether routers become neighbors by using either passive interfaces or statically defined neighbors
- Examining configuration settings that can prevent EIGRP neighborships

Manipulating EIGRP Hello and Hold Timers

The word *convergence* defines the overall process by which routers notice internetwork topology changes, communicate about those changes, and change their routing tables to contain only the best currently working routes. EIGRP converges very quickly, even with all default settings.

One of the slower components of the EIGRP convergence process relates to the timers that EIGRP neighbors use to recognize that a neighborship has failed. If the interface over which the neighbor is reachable fails, and Cisco IOS changes the interface state to anything other than "up/up," a router immediately knows that the neighborship should fail. However, in some cases, an interface state might stay "up/up" during times when the link is not usable. In such cases, EIGRP convergence relies on the Hold Timer to expire, which by default, on LANs, means a 15-second wait. (The default EIGRP Hold time on interfaces/subinterfaces with a bandwidth of T1 or lower, with an encapsulation type of Frame Relay, is 180 seconds.)

The basic operation of these two timers is relatively simple. EIGRP uses the Hello messages in part as a confirmation that the link between the neighbors still works. If a router does not receive a Hello from a neighbor for one entire Hold time, that router considers the neighbor to be unavailable. For example, with a default LAN setting of Hello = 5 and Hold = 15, the local router sends Hellos every 5 seconds. The neighbor resets its downward-counting Hold Timer to 15 upon receiving a Hello from that neighbor. Under normal operation on a LAN, with defaults, the Hold Timer for a neighbor would vary from 15, down to 10, and then be reset to 15. However, if the Hellos were no longer received for 15 seconds, the neighborship would fail, driving convergence.

To optimize convergence, an engineer could simply reduce the Hello and Hold Timers, accepting insignificant additional overhead, in return for shorter convergence times. These settings can be made per interface/subinterface, and per EIGRP process.

Note Although expected to be outside the scope of CCNP, EIGRP can also use the Bi-directional Forwarding Detection (BFD) feature, which provides a means for subsecond detection of a failure in IP connectivity between two neighboring routers.

Configuring the Hello/Hold Timers

Most design engineers would normally choose Hello/Hold Timers that match on all router interfaces on a subnet. However, these settings do not have to match. Interestingly, by setting the Hello and Hold Timers to nondefault values, you can see some oddities with how EIGRP neighbors use these values.

For example, consider four WAN distribution routers, as shown in Figure 4-3. These routers might each have a number of Frame Relay PVCs to remote branches, or multiple MPLS VPN connections to branches. However, to communicate with each other and with data centers at the home office, these four routers connect through a core VLAN/subnet. Note that the design shows routers, rather than Layer 3 switches, but the concept is the same in either case.

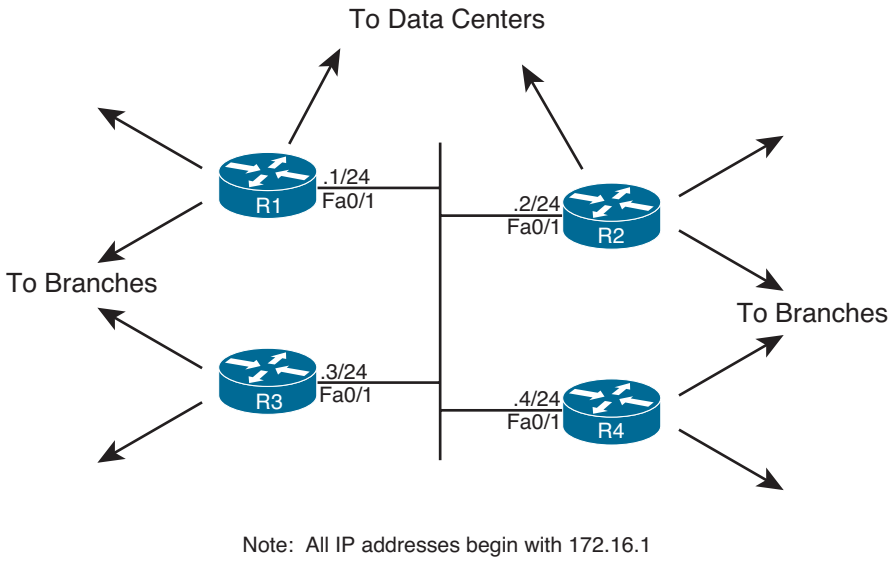


Figure 4-3 *Four WAN Distribution Routers on the Same VLAN/Subnet*

A design that hoped to speed EIGRP convergence might call for setting the Hello and Hold Timers to 2 and 6, respectively. (The Hold Timer does not have to be three times the Hello Timer, but the 3:1 ratio is a reasonable guideline.) However, to make an important point about operation of the configuration commands, Example 4-3 sets only R1's Fa0/1 timers to the new values. Note that in this case, EIGRP has already been configured on all four routers, using ASN 9.

Example 4-3 *EIGRP Hello and Hold Timer Configuration—R1*

```
interface Fastethernet0/1
 ip hello-interval eigrp 9 2
 ip hold-time eigrp 9 6
```

A couple of interesting points can be made about the operation of these seemingly simple commands. First, these two settings can be made per interface/subinterface, but not per neighbor. In Figure 4-3, the Example 4-3 configuration then applies on R1 for all three neighbors reachable on interface Fa0/1.

The second interesting point about these commands is that one parameter (the Hello Interval) tells R1 what to do, whereas the other (the Hold Timer) actually tells the neighboring routers what to do. As shown in Figure 4-4, the **ip hello-interval eigrp 9 2** interface subcommand tells R1 to send Hellos every 2 seconds. However, the **ip hold-time eigrp 9 6** interface subcommand tells R1, again for the EIGRP process with ASN 9, to tell its neighbors to use a Hold Timer of 6 for their respective neighbor relationships with R1. In short, the EIGRP Hello message sent by R1 announces the Hold Timer that other routers should use in the neighbor relationship with R1. Figure 4-4 shows this idea in graphical form.

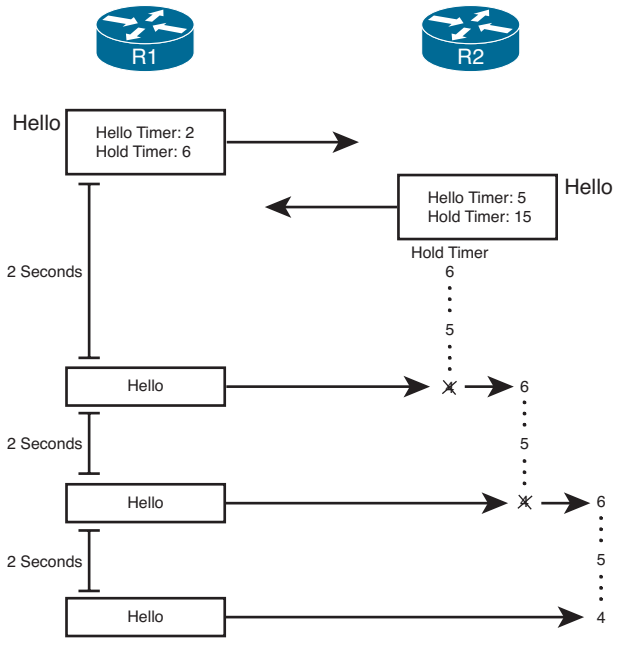


Figure 4-4 *R1 Announcing New Hello and Hold Timers*

Note Cisco IOS does not prevent you from making the unfortunate configuration choice of setting the Hold Timer to a value smaller than the Hello interval. In such a case, the neighborship repeatedly fails and recovers, flapping routes in and out of the routing table.

Verifying the Hello/Hold Timers

To find the Hello interface and Hold time configured on a router's interface, you could of course look at a router's configuration, but the **show running-config** command might not be available to you on some question types on the ROUTE exam. However, if you have access to only user mode, you can issue the **show ip eigrp interfaces detail** *type number* command. It's important to note, however, that if you use that command on some older versions of Cisco IOS, the Hold time might not displayed.

Example 4-4 shows some sample command output from R1, R2, and R3. Note that the Hello and Hold Timer settings on R1 are all in the range of 10–15 seconds, because the timers on R2, R3, and R4 all still default to 5 and 15 seconds, respectively. R2's neighborship with R1 lists a Hold Timer of 4, which is within the expected range of 4–6 seconds remaining.

Example 4-4 *Demonstration that R2 and R3 Use R1's Configured Hold Timer*

```
! On Router R1: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R1# show ip eigrp interfaces detail fa0/1

EIGRP-IPv4 Interfaces for AS(9)
                Xmit Queue   PeerQ        Mean  Pacing Time  Multicast  Pending
Interface  Peers Un/Reliable  Un/Reliable  SRTT  Un/Reliable  Flow Timer Routes
Fa0/1        3      0/0          0/0       535      0/1          50       0
  Hello-interval is 2, Hold-time is 6
  Split-horizon is enabled
  Next xmit serial <none>
  Packetized sent/expedited: 0/0
  Hello's sent/expedited: 102/1
  Un/reliable mcasts: 0/1  Un/reliable ucasts: 4/9
  Mcast exceptions: 1  CR packets: 1  ACKs suppressed: 1
  Retransmissions sent: 2  Out-of-sequence rcvd: 0
  Topology-ids on interface - 0
  Authentication mode is not set


R1# show ip eigrp neighbors
IP-EIGRP neighbors for process 9
H   Address              Interface  Hold Uptime          SRTT    RTO    Q    Seq
                                    (sec)               (ms)          Cnt  Num
2   172.16.1.4             Fa0/1        11 00:03:17      1596    5000   0    7
1   172.16.1.3             Fa0/1        11 00:05:21         1     200   0    5
0   172.16.1.2             Fa0/1        13 00:09:04         4     200   0    2
! On Router R2: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R2# show ip eigrp neighbors
IP-EIGRP neighbors for process 9
H   Address              Interface  Hold Uptime          SRTT    RTO    Q    Seq
                                    (sec)               (ms)          Cnt  Num
2   172.16.1.4            Fa0/1       11 00:03:36          4     200   0    6
1   172.16.1.3            Fa0/1       11 00:05:40         12     200   0    4
0   172.16.1.1            Fa0/1        4 00:09:22          1     200   0    2
! On Router R3: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
R3# show ip eigrp neighbors
IP-EIGRP neighbors for process 9
H   Address              Interface  Hold Uptime          SRTT    RTO     Q    Seq
                                    (sec)               (ms)           Cnt  Num
2   172.16.1.4            Fa0/1         11 00:03:40        4      200   0    5
1   172.16.1.1            Fa0/1         5 00:05:44      1278     5000   0    4
0   172.16.1.2            Fa0/1         13 00:05:44     1277     5000   0    4
```

Preventing Unwanted Neighbors Using Passive Interfaces

When an EIGRP network configuration subcommand matches an interface, EIGRP on that router does two things:

Step 1. Attempts to find potential EIGRP neighbors by sending Hellos to the 224.0.0.10 multicast address

Step 2. Advertises the subnet connected to that interface

In some cases, however, no legitimate EIGRP neighbors might exist off an interface. For example, consider the small internetwork shown in Figure 4-5, with three routers, and with only one router connected to each LAN interface. Each router needs to advertise the subnets connected to their various FastEthernet interfaces, but at the same time, there is no benefit to multicast EIGRP Hellos on those interfaces, because only one router connects to each LAN.

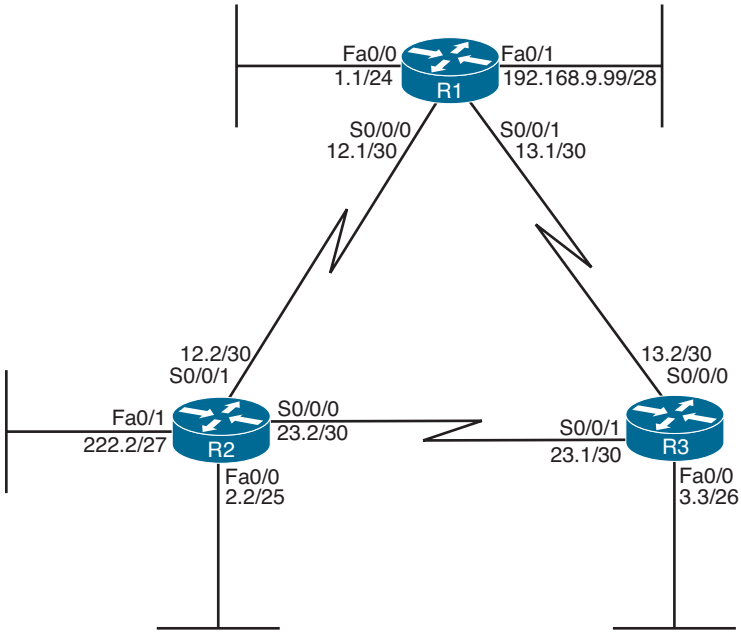


Figure 4-5 *LAN Interfaces That Benefit from the Passive Interface Feature*

The network designer can reasonably choose to limit EIGRP on those interfaces that have no legitimate EIGRP neighbors. However, the subnets connected to those same interfaces also typically need to be advertised by EIGRP. For example, subnet 10.1.1.0/24, off R1's Fa0/0 interface, still needs to be advertised by EIGRP, even though R1 should never find an EIGRP neighbor on that interface.

Given such a requirement—to advertise the subnet while disallowing EIGRP neighborships on the interface—an engineer has two main configuration options to choose from:

- Enable EIGRP on the interface using the EIGRP **network** command, but tell the router to not send any EIGRP messages on the interface by making the interface passive (using the **passive-interface** command).
- Do not enable EIGRP on the interface, and advertise the connected route using route redistribution (and the **redistribute connected** configuration command).

The first option relies on the passive interface feature—a feature specifically created with this design requirement in mind. When an interface is passive, EIGRP does not send any EIGRP messages on the interface—multicasts or EIGRP unicasts—and the router ignores any EIGRP messages received on the interface. However, EIGRP still advertises the connected subnets if matched with an EIGRP **network** command. As a result, the first option in the preceding list directly meets all the design requirements. It has the added advantage of being very secure in that no EIGRP neighborships are possible on the interface.

The second option—redistributing connected subnets—also works, but frankly it is the less preferred option in this case. Specifically, the passive interface option clearly meets the design requirements, while the redistribution option causes the connected route to be advertised as an external EIGRP route. This could cause problems in some cases with multiple redistribution points between routing domains (as discussed in Chapter 10, "Route Redistribution").

The configuration of the passive interface itself is fairly straightforward. To configure the passive interface option, these three routers could be configured as shown in Example 4-5.

Example 4-5 *Configuration of* **passive-interface** *Commands on R1, R2, and R3*

```
! On Router R1: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router eigrp 1
 passive-interface fastethernet0/0
 passive-interface fastethernet0/1
 network 10.0.0.0
 network 192.168.9.0
```

```
! On Router R2: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router eigrp 1
 passive-interface default
 no passive-interface serial0/0/0
 no passive-interface serial0/0/1
 network 10.0.0.0
```

```
! On Router R3: !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
router eigrp 1
 passive-interface fastethernet0/0
 network 10.0.0.0
```

R1's configuration lists two **passive-interface** commands, one per LAN interface. As a result, R1 no longer sends EIGRP messages on these two interfaces, including the multicast EIGRP Hellos used to discover neighbors.

R2's configuration uses a slightly different option: the **passive-interface default** command. This command essentially changes the default for an interface from not being passive to instead being passive. Then, to make an interface not passive, you have to use a **no** version of the **passive-interface** command for those interfaces.

Two commands help to verify that the passive interface design is working properly. First, the **show ip eigrp interfaces** command omits passive interfaces, listing the nonpassive interfaces matched by a **network** command. Alternatively, the **show ip protocols** command explicitly lists all passive interfaces. Example 4-6 shows samples of both commands on R2.

Example 4-6 *Verifying the Results of* **passive-interface** *on R2*

```
R2# show ip eigrp interfaces
IP-EIGRP interfaces for process 1


                      Xmit Queue   Mean     Pacing Time     Multicast   Pending
Interface      Peers  Un/Reliable  SRTT     Un/Reliable     Flow Timer  Routes
Se0/0/0            1       0/0        32        0/15             159        0
Se0/0/1            1       0/0      1290        0/15            6443        0
R2# show ip protocols
Routing Protocol is "eigrp 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=1, K2=0, K3=1, K4=0, K5=0
  EIGRP maximum hopcount 100
  EIGRP maximum metric variance 1
  Redistributing: eigrp 1
  EIGRP NSF-aware route hold timer is 240s
  Automatic network summarization is in effect
  Maximum path: 4
  Routing for Networks:
    10.0.0.0
  Passive Interface(s):
    FastEthernet0/0
    FastEthernet0/1
  Routing Information Sources:
    Gateway         Distance      Last Update
    10.1.12.1              90      00:00:39
    10.1.23.1              90      00:00:39
  Distance: internal 90 external 170
```

Controlling Neighborships with Static Configuration

EIGRP supports the ability to statically define neighbors instead of dynamically discovering neighbors.

Although seldom used, you can use this feature to reduce the overhead associated with EIGRP multicast messages. Frame Relay WANs in particular might benefit from the static neighbor definitions, because to support multicasts and broadcasts over Frame Relay, a router must replicate a frame and send a copy over every PVC associated with the interface or subinterface. For example, if a multipoint subinterface has ten PVCs associated with it, but only two of the remote routers used EIGRP, without static neighbors, all ten routers would be sent a copy of the EIGRP multicast Hello packets. With static neighbor definitions for the two routers, EIGRP messages would be sent as unicasts to each of the two neighbors, with no EIGRP messages sent to the eight non-EIGRP routers, reducing overhead.

The configuration seems simple, but it has a few subtle caveats. This section examines the straightforward configuration first and then examines the caveats.

Configuring Static EIGRP Neighbors

To define a neighbor, both routers must configure the **neighbor** *ip-address outgoing-interface* EIGRP router subcommand. The IP address is the interface IP address of the neighboring router. Also, the configured IP address must be from the subnet connected to the interface listed in the **neighbor** command; otherwise, the command is rejected. Also, note that the EIGRP configuration still needs a **network** command that matches the interface referenced by the **neighbor** command.

For example, consider Figure 4-6, which adds a new router (R5) to the internetwork of Figure 4-3. R1 and R5 have a PVC connecting them, with IP addresses and subinterface numbers shown.

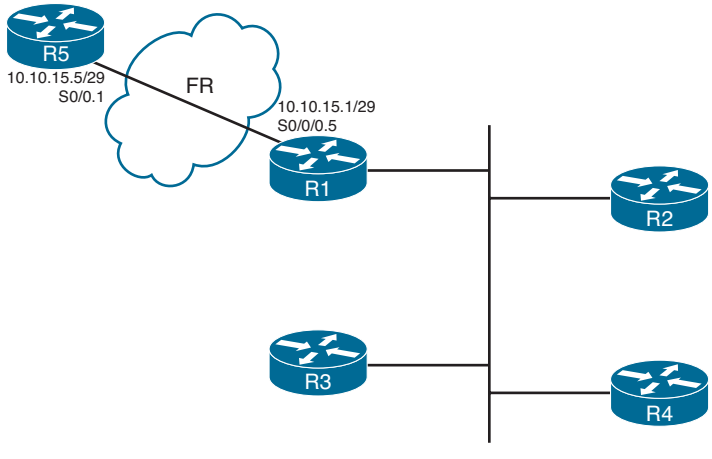


Figure 4-6 *Adding a Branch, with a Static EIGRP Neighbor*

Example 4-7 shows the configuration on both R1 and R5 to use static neighbor definitions. Of note, R1's neighbor command refers to R5's IP address on their common subnet (10.10.15.5), with R1's local interface (S0/0/0.5). R5 lists the reverse, with R1's 10.10.15.1 IP address and R5's local S0/0.1 interface. Also note that both routers have a **network** command that references network 10.0.0.0, and both routers do advertise subnet 10.10.15.0/29.

The **show ip eigrp neighbors** command does not identify a neighbor as static, but the **show ip eigrp neighbors detail** command does. Example 4-7 shows the more detailed output near the end, with the designation of 10.10.15.5 (R5) as a static neighbor.

Example 4-7 *Static EIGRP Neighborship Between R1 and R5*

```
! New configuration on router R1
R1# show running-config
! lines omitted
router eigrp 9
 network 172.16.0.0
 network 10.0.0.0
 no auto-summary
 neighbor 10.10.15.5 Serial0/0/0.5
! Back to R1
R1# show ip eigrp neighbors detail
IP-EIGRP neighbors for process 9
H   Address                 Interface    Hold Uptime  SRTT  RTO      Q        Seq
                                           (sec)      (ms)          Cnt      Num
3   10.10.15.5              Se0/0/0.5   10 00:00:51     15  200      0        2
   Static neighbor
   Version 12.4/1.2, Retrans: 0, Retries: 0
2   172.16.1.2              Fa0/1        11 00:02:57    3  200        0       25
   Version 12.4/1.2, Retrans: 1, Retries: 0
1   172.16.1.3              Fa0/1        10 00:03:45    5  200        0       21
   Version 12.4/1.2, Retrans: 0, Retries: 0
0   172.16.1.4              Fa0/1        13 00:03:45    5  200        0       18
```

```
! R5's new config added to support the neighbor
R5# show running-config
! lines omitted
router eigrp 9
 network 10.0.0.0
 no auto-summary
 neighbor 10.10.15.1 Serial0/0.1
```

Caveat When Using EIGRP Static Neighbors

Cisco IOS changes how it processes EIGRP packets on any interface referenced by an EIGRP **neighbor** command. Keeping in mind the design goal for this feature—to reduce

multicasts—Cisco IOS disables all EIGRP multicast packet processing on an interface when an EIGRP **neighbor** command has been configured. For example, in Example 4-7, R1's S0/0/0.5 subinterface will not process EIGRP multicast packets any more as a result of R1's **neighbor 10.10.15.5 Serial0/0/0.5** EIGRP subcommand.

Because of the operation of the EIGRP **neighbor** command, if at least one EIGRP static neighbor is defined on an interface, no dynamic neighbors can be either discovered or continue to work if already discovered. For example, again in Figure 4-6 and Example 4-7, if R1 added a **neighbor 172.16.1.5 FastEthernet0/1** EIGRP subcommand, R1 would lose its current neighborships with Routers R2, R3, and R4.

Configuration Settings That Could Prevent Neighbor Relationships

Some of the configuration settings already mentioned in this chapter, when configured incorrectly, might prevent EIGRP neighborships. This section summarizes those settings, and introduces a few other configuration settings that can prevent neighbor relationships. The list of items that must match—and that do not have to match—can be a useful place to start troubleshooting neighbor initialization problems in real life, and to troubleshoot neighborship problems for simulation questions on the CCNP ROUTE exam.

Table 4-4 lists the neighbor requirements for both EIGRP and Open Shortest Path First (OSPF). (OSPF is included here just as a frame of reference for those more familiar with OSPF; this information will be repeated in Chapter 7, "Fundamental OSPF Concepts," which discusses OSPF neighborship requirements.) Following the table, the next few pages examine some of these settings for EIGRP.

Table 4-4 *Neighbor Requirements for EIGRP and OSPF*

Requirement	EIGRP	OSPF
The routers must be able to send/receive IP packets to one another.	Yes	Yes
Interfaces' primary IP addresses must be in same subnet.	Yes	Yes
Must not be passive on the connected interface.	Yes	Yes
Must use the same ASN (EIGRP) or process-ID (OSPF) in the router configuration command.	Yes	No
Hello interval/timer, plus either the Hold (EIGRP) or Dead (OSPF) timer, must match.	No	Yes
Must pass neighbor authentication (if configured).	Yes	Yes
Must be in same area.	N/A	Yes
IP MTU must match.	No	Yes
K-values (used in metric calculation) must match.	Yes	—
Router IDs must be unique.	No[1]	Yes

[1] Duplicate EIGRP RIDs do not prevent routers from becoming neighbors, but it can cause problems when adding external EIGRP routes to the IP routing table.

Going through Table 4-4 sequentially, the first two items relate to IP connectivity. Two routers must be able to send and receive IP packets with each other. Additionally, the primary IP address on the interfaces—in other words, the IP address configured without the **secondary** keyword on the **ip address** command—must be in the same subnet.

Note It should not matter for CCNP ROUTE, but possibly for CCIE R/S: EIGRP's rules about neighbor IP addresses being in the same subnet are less exact than OSPF. OSPF requires matching subnet numbers and masks. EIGRP just asks the question of whether the neighbor's IP address is in the range of addresses for the subnet as known to the local router. For example, two routers with addresses of 10.1.1.1/24 (range 10.1.1.1–10.1.1.254) and 10.1.1.2/30 (range 10.1.1.1–10.1.1.2) would actually allow EIGRP neighborship, because each router believes the neighbor's IP address to be in the same subnet as the local router.

The next three items in Table 4-4—passive interfaces, matching the EIGRP ASN number, and allowing mismatching Hello/Hold Timers—have already been covered in this chapter.

The next item, authentication, is discussed in detail in Chapter 17, "Routing Protocol Authentication."

The next two items in the table—matching the IP MTU and matching OSPF areas—do not prevent EIGRP neighborships. These topics, are requirements for OSPF neighborship and will be discussed in Chapter 7.

Finally, the last two items in the table (K-values and router IDs) each require more than a cursory discussion for EIGRP and will be explained in the upcoming pages.

Configuring EIGRP Metric Components (K-values)

EIGRP calculates its integer metric, by default, using a formula that uses constraining bandwidth and cumulative delay. You can change the formula to use link reliability and link load, and even disable the use of bandwidth and/or delay. To change the formula, an engineer can configure five weighting constants, called K-values, which are represented in the metric calculation formula as constants K1, K2, K3, K4, and K5.

From a design perspective, Cisco strongly recommends against using link load and link reliability in the EIGRP metric calculation. Most shops that use EIGRP never touch the K-values at all. However, in labs, it can be useful to disable the use of bandwidth from the metric calculation, because that simplifies the metric math and makes it easier to learn the concepts behind EIGRP.

The **metric weights** command sets five variables (K1 through K5), each of which weights the metric calculation formula more or less heavily for various parts of the formula. Mismatched K-value settings prevent two routers from becoming neighbors. Thankfully, determining whether such a mismatch exists is easy. When a router receives an EIGRP Hello with mismatched K-values (as compared to itself), the router issues a log message

stating that a K-value mismatch exists. You can also examine the values either by looking at the running configurations or by looking for the K-values listed in the output of the **show ip protocols** command, as shown in Example 4-8.

Note In the command **metric weights 0 1 0 1 1 0**, the first number (that is, the left-most 0) represents the Type of Service (ToS) value with which EIGRP packets should be marked. This is a Quality of Service (QoS) setting. It equals 0 and cannot be changed to a different value. The remaining five numbers are the K-values: K1, K2, K3, K4, and K5, respectively.

Example 4-8 *Mismatched K-values*

```
R2(config)# router eigrp 1
R2(config-router)# metric weights 0 1 0 1 1 0
R2(config-router)# end
Feb 23 18:48:21.599: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.12.1
  (Serial0/0/1) is down: metric changed
R2#
Feb 23 18:48:24.907: %DUAL-5-NBRCHANGE: IP-EIGRP(0) 1: Neighbor 10.1.12.1
  (Serial0/0/1) is down: K-value mismatch
R2# show ip protocols
Routing Protocol is "eigrp 1"
  Outgoing update filter list for all interfaces is not set
  Incoming update filter list for all interfaces is not set
  Default networks flagged in outgoing updates
  Default networks accepted from incoming updates
  EIGRP metric weight K1=1, K2=0, K3=1, K4=1, K5=0
! lines omitted for brevity
```

EIGRP Router ID

EIGRP uses a concept of a representing each router with a router ID (RID). The EIGRP RID is a 32-bit number, represented in dotted decimal. Each router determines its RID when the EIGRP process starts, using the same general rules as does OSPF for determining the OSPF RID, as follows:

Step 1. Use the configured value (using the **eigrp router-id** *a.b.c.d* EIGRP subcommand).

Step 2. Use the highest IPv4 address on an up/up loopback interface.

Step 3. Use the highest IPv4 address on an up/up nonloopback interface.

Although EIGRP does require each router to have an RID, the actual value is of little practical importance. The EIGRP **show** commands seldom list the RID value, and unlike

OSPF RIDs, engineers do not need to know each router's EIGRP RID to interpret the EIGRP topology database. Additionally, although it is best to make EIGRP RIDs unique, duplicate RIDs do not prevent routers from becoming neighbors.

The only time the value of EIGRP RIDs matters is when injecting external routes into EIGRP. In that case, the routers injecting the external routes must have unique RIDs to avoid confusion.

Neighborship over WANs

EIGRP configuration and neighborship rules do not differ when comparing typical LAN and typical WAN technologies. However, some design and operational differences exist, particularly regarding which routers become neighbors with which other routers. This short section closes the EIGRP neighbor discussion with a brief look at Frame Relay, MPLS VPNs, and Metro Ethernet as implemented with Virtual Private LAN Service (VPLS).

Neighborship on Frame Relay

Frame Relay provides a Layer 2 WAN service. Each router connects to the service using a physical serial link, called a Frame Relay access link. The provider then creates logical connections, called *permanent virtual circuits (PVC)*, which are logical paths between pairs of routers connected to a Frame Relay service. Any pair of routers that connect to the ends of a Frame Relay PVC can send Frame Relay frames to each other. Therefore, they can send IP packets and become EIGRP neighbors. Figure 4-7 shows a typical case, with R1 as a central-site router, and R2, R3, and R4 acting as branch routers.

Figure 4-7 shows EIGRP neighborships, but note that all routers can learn all routes in the internetwork, even though not all routers become neighbors. The neighborships can only form when a PVC exists between the two routers.

Neighborship on MPLS VPN

Multiprotocol Label Switching (MPLS) Virtual Private Networks (VPN) create a WAN service that has some similarities but many differences when compared to Frame Relay. The customer routers connect to the service, often with serial links but at other times with Frame Relay PVCs or with Ethernet. The service itself is a Layer 3 service, forwarding IP packets through a cloud. As a result, no predefined PVCs need to exist between the customer routers. Additionally, the service uses routers at the edge of the service provider cloud—generically called *provider edge (PE)* routers—and these routers are Layer 3 aware.

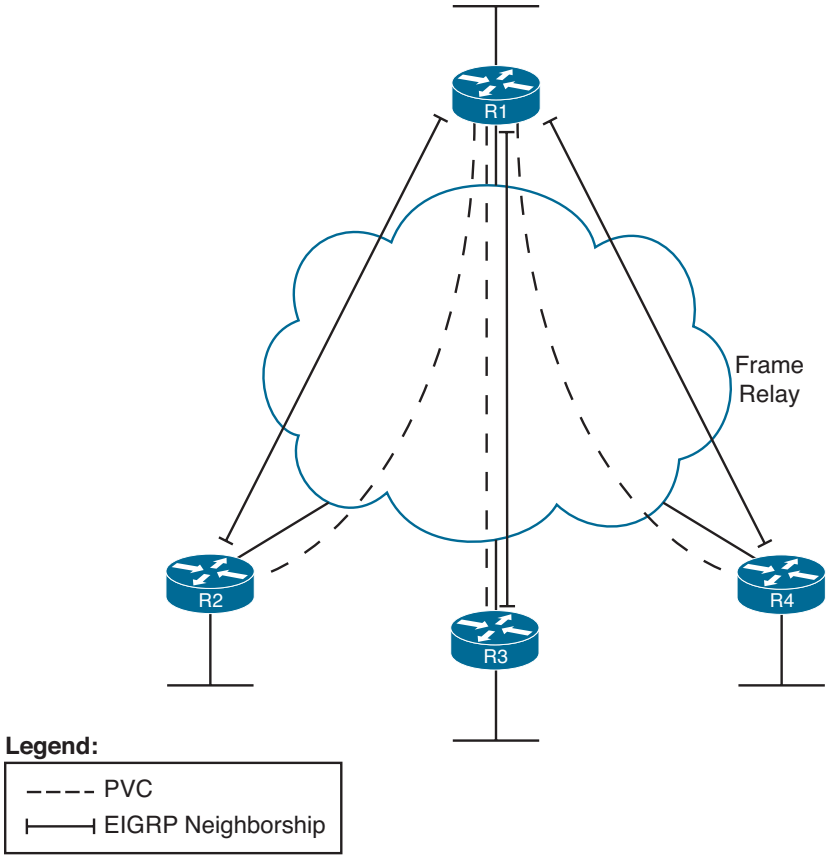


Figure 4-7 *EIGRP Neighborships over Frame Relay*

That Layer 3 awareness means that the *customer edge (CE)* routers form an EIGRP neighborship with the PE router on the other end of their local access link, as shown in Figure 4-8. The PE routers exchange their routes, typically using Multiprotocol BGP (MP-BGP), a topic outside the scope of this book. However, all the CE routers then learn routes from each other, although each CE router has only one EIGRP neighborship for each of its connections into the MPLS VPN cloud.

Neighborship on Metro Ethernet

The term *Metropolitan Ethernet (MetroE)* represents a range of Layer 2 WAN services in which the CE device connects to the WAN service using some form of Ethernet. Because MetroE provides a Layer 2 Ethernet service, the service delivers an Ethernet frame sent by one customer router to another customer router (for unicast frames), or to many other routers (for multicast or broadcast frames).

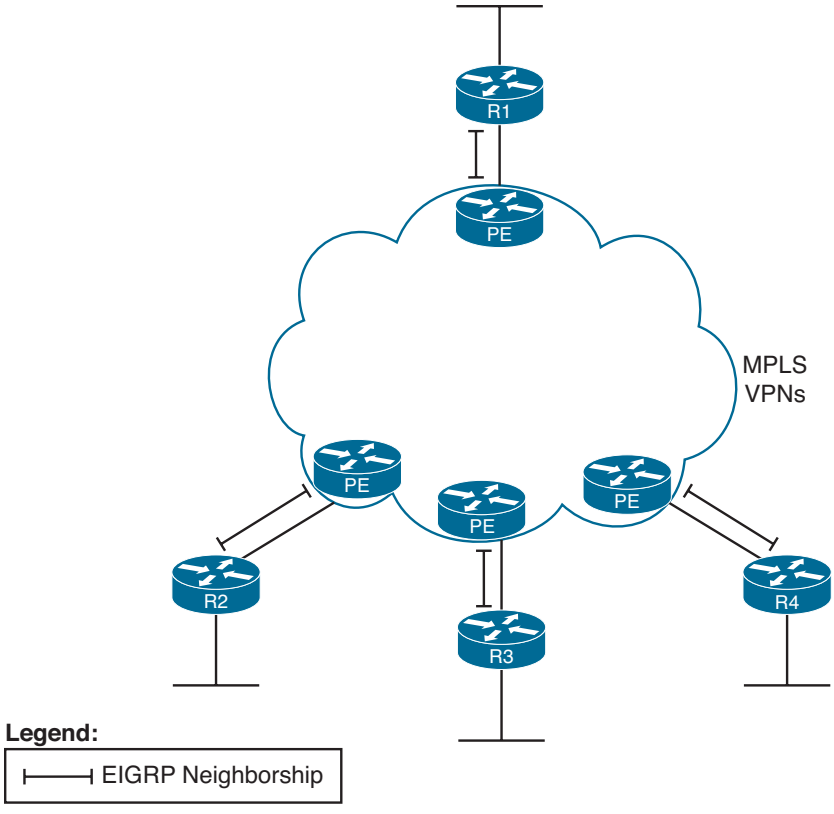


Figure 4-8 *EIGRP Neighborships over MPLS VPN*

MetroE encompasses several underlying technologies to create the service. Of note for the purposes of this book are the Virtual Private Wire Service (VPWS) and the Virtual Private LAN Service (VPLS). Both technical specifications allow for connections using Ethernet links, with the service forwarding Ethernet frames. VPWS focuses on point-to-point topologies, whereas VPLS supports multipoint, approximating the concept of the entire WAN service acting like one large Ethernet switch. Because it is a Layer 2 service, MetroE does not have any Layer 3 awareness, and customer routers (typically referenced with the more general service provider term *customer premises equipment*, or *CPE*) see the MetroE service as a VLAN. Because the customer routers connect to the service as a VLAN, all the routers connected to the service can become EIGRP neighbors, as shown in Figure 4-9.

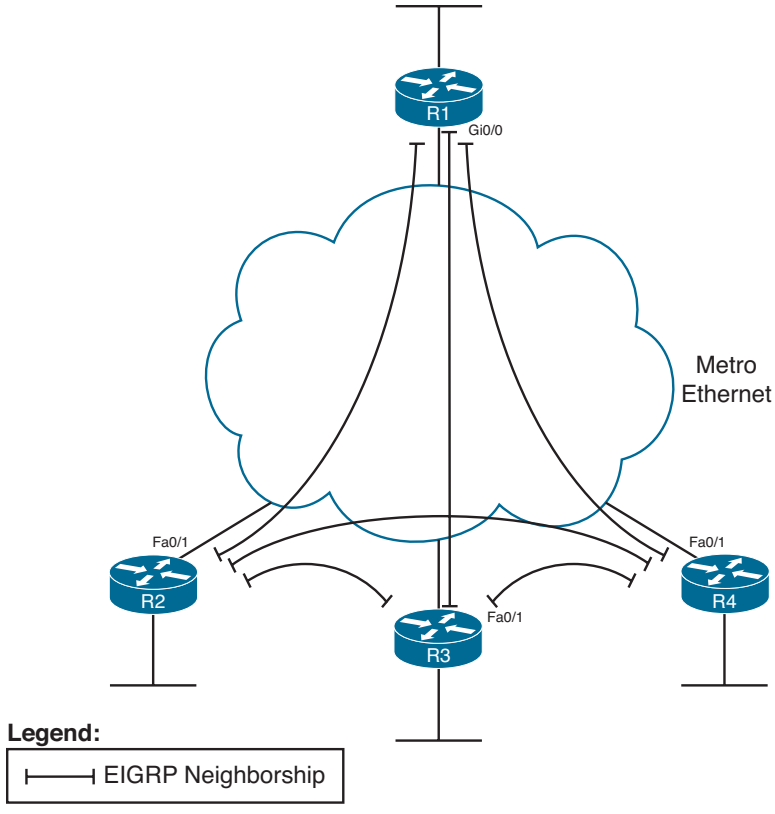


Figure 4-9 *EIGRP Neighborships over Metro Ethernet*

Exam Preparation Tasks

Planning Practice

The CCNP ROUTE exam expects test takers to be able to review design documents, create implementation plans, and create verification plans. This section provides some exercises that can help you to take a step back from the minute details of the topics in this chapter, so that you can think about the same technical topics from the planning perspective.

For each planning practice table, simply complete the table. Note that any numbers in parentheses represent the number of options listed for each item in the solutions in Appendix F, "Completed Planning Practice Tables," which you can find on the CD-ROM accompanying this book.

Design Review Table

Table 4-5 lists several design goals related to this chapter. If these design goals were listed in a design document, and you had to take that document and develop an implementation plan, what implementation options come to mind? For any configuration items, a general description can be used, without any concern about the specific parameters.

Table 4-5 *Design Review*

Design Goal	Possible Implementation Choices Covered in This Chapter
Improve EIGRP convergence.	
Implement EIGRP on each router so that neighborships are formed (2).	
Limit neighborship formation on interfaces matched with an EIGRP **network** command (3).	

Implementation Plan Peer Review Table

Table 4-6 shows a list of questions that others might ask, or that you might think about, during a peer review of another network engineer's implementation plan. Complete the table by answering the questions.

Table 4-6 *Notable Questions from This Chapter to Consider During an Implementation Plan Peer Review*

Question	Answer
What happens on a router interface on which an EIGRP **network** command matches the interface? (2)	
What configuration settings prevent EIGRP neighbor discovery on an EIGRP-enabled interface? (2)	
What configuration settings prevent any neighborships on an EIGRP-enabled interface?	
What settings do potential neighbors check before becoming EIGRP neighbors? (5)	
What settings that you might think would impact EIGRP neighbor relationships actually do not prevent neighborship? (3)	

Create an Implementation Plan Table

To practice skills useful when creating your own EIGRP implementation plan, list in Table 4-7 configuration commands related to the configuration of the following features. You might want to record your answers outside the book, and set a goal to complete this table (and others like it) from memory during your final reviews before taking the exam.

Table 4-7 *Implementation Plan Configuration Memory Drill*

Feature	Configuration Commands/Notes
Enabling EIGRP on interfaces	
Setting Hello and Hold Timers	
Passive interfaces	
Static EIGRP neighbors	
K-values	
EIGRP router ID	

Choose Commands for a Verification Plan Table

To practice skills useful when creating your own EIGRP verification plan, list in Table 4-8 all commands that supply the requested information. You might want to record your answers outside the book, and set a goal to complete this table (and others like it) from memory during your final reviews before taking the exam.

Table 4-8 *Verification Plan Memory Drill*

Information Needed	Command
Routes that have been added to the IP routing table by EIGRP.	
All routes in a router's routing table.	
The specific route for a single destination address or subnet.	
A listing of all (both statically configured and dynamically discovered) EIGRP neighbors.	
Notation as to whether a neighbor was dynamically discovered or statically configured.	
A listing of statistics regarding the numbers of EIGRP messages sent and received by a router.	
A listing of interfaces on which EIGRP has been enabled (by virtue of the EIGRP **network** command).	
A listing of the number of EIGRP peers known through a particular interface.	
The elapsed time since a neighborship was formed.	
The parameters of any EIGRP **network** commands.	
The configured Hello Timer for an interface.	
The configured Hold Timer for an interface.	
The current actual Hold Timer for a neighbor.	
A router's EIGRP ASN.	
A list of EIGRP passive interfaces.	
A list of nonpassive EIGRP interfaces.	
A listing of EIGRP K-values.	
A listing of traffic statistics about EIGRP.	
A router's EIGRP Router ID.	

Review All the Key Topics

Review the most important topics from inside the chapter, noted with the Key Topic icon in the outer margin of the page. Table 4-9 lists a reference of these key topics and the page numbers on which each is found.

Table 4-9 *Key Topics for Chapter 4*

Key Topic Element	Description	Page Number
List	Configuration step review for basic EIGRP configuration	7
Table 4-2	Key EIGRP verification commands	10
Table 4-3	Summary of EIGRP features and facts	15
List	Methods of disallowing EIGRP neighborships on an interface, while still advertising the connected subnet	22
Table 4-4	List of items that can impact the formation of EIGRP neighborships	26
List	Rules for choosing an EIGRP Router ID	28

Complete the Tables and Lists from Memory

Print a copy of Appendix D, "Memory Tables," (found on the CD), or at least the section for this chapter, and complete the tables and lists from memory. Appendix E, "Memory Tables Answer Key," also on the CD, includes completed tables and lists to check your work.

Define Key Terms

Define the following key terms from this chapter, and check your answers in the glossary:

K-value, neighborship, Hello interval, Hold Timer, passive interface

CISCO

Official Cert Guide

Learn, prepare, and practice for exam success

CCNP

Routing and Switching SWITCH 300-115

ciscopress.com

DAVID HUCABY, CCIE® No. 4594

Contents

This chapter covers the following topics that you need to master for the CCNP SWITCH exam:

- **Layer 2 Switch Operation:** This section describes the functionality of a switch that forwards Ethernet frames.
- **Multilayer Switch Operation:** This section describes the mechanisms that forward packets at OSI Layers 3 and 4.
- **Tables Used in Switching:** This section explains how tables of information and computation are used to make switching decisions. Coverage focuses on the content-addressable memory table involved in Layer 2 forwarding, and the ternary content-addressable memory used in packet-handling decisions at Layers 2 through 4.
- **Managing Switching Tables:** This section reviews the Catalyst commands that you can use to configure and monitor the switching tables and memory. These commands can be useful when troubleshooting or tracing the sources of data or problems in a switched network.

CHAPTER 2

Switch Operation

To have a good understanding of the many features that you can configure on a Catalyst switch, you first should understand the fundamentals of the switching function.

This chapter serves as a primer, describing how an Ethernet switch works. It presents Layer 2 forwarding, along with the hardware functions that make forwarding possible. Multilayer switching is also explained. A considerable portion of the chapter deals with the memory architecture that performs switching at Layers 3 and 4 both flexibly and efficiently. This chapter also provides a brief overview of useful switching table management commands.

"Do I Know This Already?" Quiz

The "Do I Know This Already?" quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the "Exam Preparation Tasks" section. If you are in doubt based on your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. Table 2-1 outlines the major headings in this chapter and the "Do I Know This Already?" quiz questions that go with them. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

Table 2-1 *"Do I Know This Already?" Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions Covered in This Section
Layer 2 Switch Operation	1–5
Multilayer Switch Operation	6–9
Switching Tables	10–11
Troubleshooting Switching Tables	12

1. Which of the following devices performs transparent bridging?
   - **a.** Ethernet hub
   - **b.** Layer 2 switch
   - **c.** Layer 3 switch
   - **d.** Router

2. When a PC is connected to a Layer 2 switch port, how far does the collision domain spread?
   a. No collision domain exists.
   b. One switch port.
   c. One VLAN.
   d. All ports on the switch.
3. What information is used to forward frames in a Layer 2 switch?
   a. Source MAC address
   b. Destination MAC address
   c. Source switch port
   d. IP addresses
4. What does a switch do if a MAC address cannot be found in the CAM table?
   a. The frame is forwarded to the default port.
   b. The switch generates an ARP request for the address.
   c. The switch floods the frame out all ports (except the receiving port).
   d. The switch drops the frame.
5. In a Catalyst switch, frames can be filtered with access lists for security and QoS purposes. This filtering occurs according to which of the following?
   a. Before a CAM table lookup
   b. After a CAM table lookup
   c. Simultaneously with a CAM table lookup
   d. According to how the access lists are configured
6. Access list contents can be merged into which of the following?
   a. CAM table
   b. TCAM table
   c. FIB table
   d. ARP table
7. Multilayer switches using CEF are based on which of these techniques?
   a. Route caching
   b. Netflow switching
   c. Topology-based switching
   d. Demand-based switching

8. Which answer describes multilayer switching with CEF?
   a. The first packet is routed and then the flow is cached.
   b. The switch supervisor CPU forwards each packet.
   c. The switching hardware learns station addresses and builds a routing database.
   d. A single database of routing information is built for the switching hardware.
9. In a switch, frames are placed in which buffer after forwarding decisions are made?
   a. Ingress queues
   b. Egress queues
   c. CAM table
   d. TCAM
10. What size are the mask and pattern fields in a TCAM entry?
   a. 64 bits
   b. 128 bits
   c. 134 bits
   d. 168 bits
11. Access list rules are compiled as TCAM entries. When a packet is matched against an access list, in what order are the TCAM entries evaluated?
   a. Sequentially in the order of the original access list.
   b. Numerically by the access list number.
   c. Alphabetically by the access list name.
   d. All entries are evaluated in parallel.
12. Which Catalyst IOS command can you use to display the addresses in the CAM table?
   a. **show cam**
   b. **show mac address-table**
   c. **show mac**
   d. **show cam address-table**

Foundation Topics

Layer 2 Switch Operation

Consider a simple network that is built around many hosts that all share the same available bandwidth. This is known as a shared media network and was used in early legacy LANs made up of Ethernet hubs. The carrier sense multiple access collision detect (CSMA/CD) scheme determines when a device can transmit data on the shared LAN.

Key Topic

When more than one host tries to talk at one time, a collision occurs, and everyone must back off and wait to talk again. This forces every host to operate in half-duplex mode, by either talking *or* listening at any given time. In addition, when one host sends a frame, all connected hosts hear it. When one host generates a frame with errors, everyone hears that, too. This type of LAN is a *collision domain* because all device transmissions are susceptible to collisions.

An Ethernet switch operates at OSI Layer 2, making decisions about forwarding frames based on the destination MAC addresses found within the frames. This means that the Ethernet media is no longer shared among connected devices. Instead, at its most basic level, an Ethernet switch provides isolation between connected hosts in several ways:

- The collision domain's scope is severely limited. On each switch port, the collision domain consists of the switch port itself and the devices directly connected to that port—either a single host or, if a shared-media hub is connected, the set of hosts connected to the hub.
- Host connections can operate in full-duplex mode because there is no contention on the media. Hosts can talk *and* listen at the same time.
- Bandwidth is no longer shared. Instead, each switch port offers dedicated bandwidth across a switching fabric to another switch port. (These frame forwarding paths change dynamically.)
- Errors in frames are not propagated. Each frame received on a switch port is checked for errors. Good frames are regenerated when they are forwarded or transmitted. This is known as *store-and-forward* switching technology: Packets are received, stored for inspection, and then forwarded.
- You can limit broadcast traffic to a volume threshold.
- Other types of intelligent filtering or forwarding become possible.

Transparent Bridging

A Layer 2 switch is basically a multiport transparent bridge, where each switch port is its own Ethernet LAN segment, isolated from the others. Frame forwarding is based completely on the MAC addresses contained in each frame, such that the switch will not forward a frame unless it knows the destination's location. (When the switch does not know

where the destination is, it makes some safe assumptions.) Figure 2-1 shows the progression from a two-port to a multiport transparent bridge, and then to a Layer 2 switch.

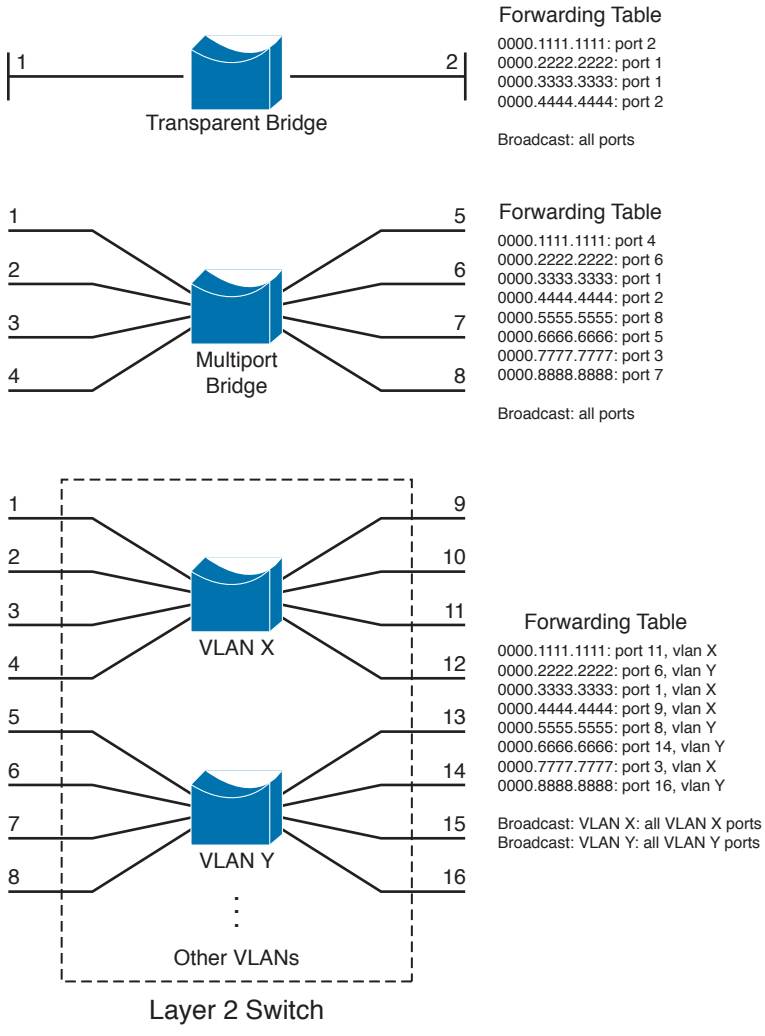


Figure 2-1 *A Comparison of Transparent Bridges and Switches*

The entire process of forwarding Ethernet frames then becomes figuring out what MAC addresses connect to which switch ports. For example, the Layer 2 switch in Figure 2-1 knows that the device using MAC address 0000.5555.5555 is located on switch port 8, which is assigned to VLAN Y. It also knows that frames arriving on VLAN Y and destined for the broadcast MAC address must be flooded out all ports that are assigned to VLAN Y.

A switch either must be told explicitly where hosts are located or must learn this information for itself. You can configure MAC address locations through a switch's command-line interface, but this quickly gets cumbersome when there are many stations on the network or when stations move around from one switch port to another.

To dynamically learn about station locations, a switch listens to incoming frames and keeps a table of address information. In Figure 2-1, this information is kept in a forwarding table. As a frame is received on a switch port, the switch inspects the source MAC address. If that address is not in the address table already, the MAC address, switch port, and virtual LAN (VLAN) on which it arrived are recorded in the table. Learning the address locations of the incoming packets is easy and straightforward.

Key Topic

Incoming frames also include the destination MAC address. Again, the switch looks up this address in the address table, hoping to find the switch port and VLAN where the destination address is attached. If it is found, the frame can be forwarded out the corresponding switch port. If the address is not found in the table, the switch must take more drastic action—the frame is forwarded in a "best effort" fashion by flooding it out all switch ports assigned to the source VLAN. This is known as unknown unicast flooding, because the location of the unicast destination is unknown.

Figure 2-2 illustrates this process, using only a single VLAN for simplification. Suppose a packet arrives on switch port 3, containing destination MAC address 0000.aaaa.aaaa. The switch looks for that MAC address in its forwarding table, but is unable to find a matching entry. The switch then floods copies of the packet out every other port that is assigned to port 3's VLAN, to increase the likelihood that 0000.aaaa.aaaa will eventually receive the packet that is destined for it. If the destination is the broadcast MAC address, the switch knows that the frame should be flooded out all ports on the VLAN.

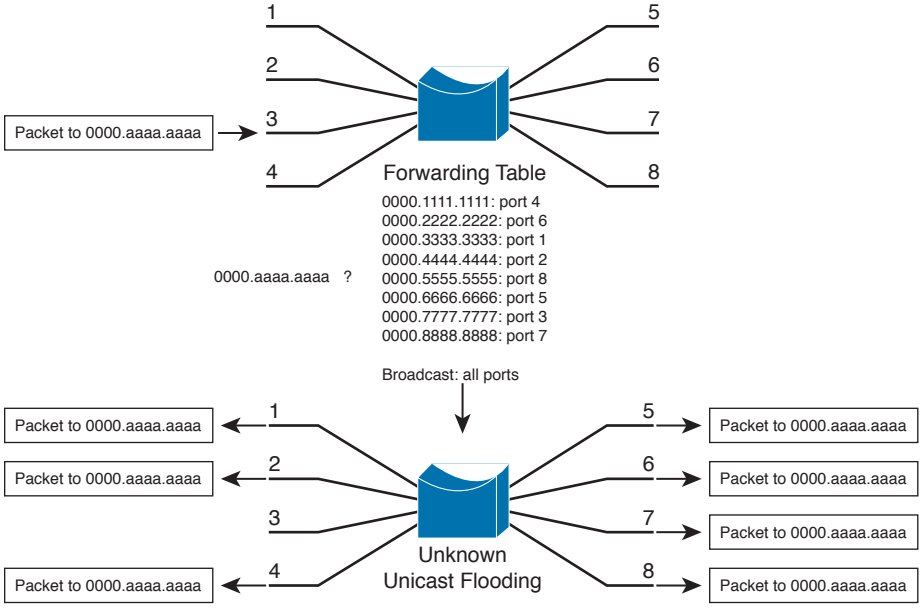


Figure 2-2 *Unknown Unicast Flooding*

A switch constantly listens to incoming frames on each of its ports, learning source MAC addresses. However, be aware that the learning process is allowed only when the Spanning Tree Protocol (STP) algorithm has decided that a port is stable for normal use. STP is concerned only with maintaining a loop-free network, where frames will not be forwarded recursively. If a loop formed, a flooded frame could follow the looped path, where it would be flooded again and again. STP is covered in greater detail in Chapters 8 through 11.

In a similar manner, frames containing a broadcast or multicast destination address are also flooded. These destination addresses are not unknown—the switch knows them well because they use standardized address values. For example, the Ethernet broadcast address is always ffff.ffff.ffff, IPv4 multicast addresses always begin with 01xx.xxxx.xxxx, and IPv6 multicast addresses begin with 3333.xxxx.xxxx. These addresses are destined for multiple locations, so they must be flooded by definition. In the case of multicast addresses, flooding is performed by default unless more specific recipient locations have been learned.

Follow That Frame!

You should have a basic understanding of the operations that a frame undergoes as it passes through a Layer 2 switch. This helps you get a firm grasp on how to configure the switch for complex functions. Figure 2-3 shows a typical Layer 2 Catalyst switch and the decision processes that take place to forward each frame.

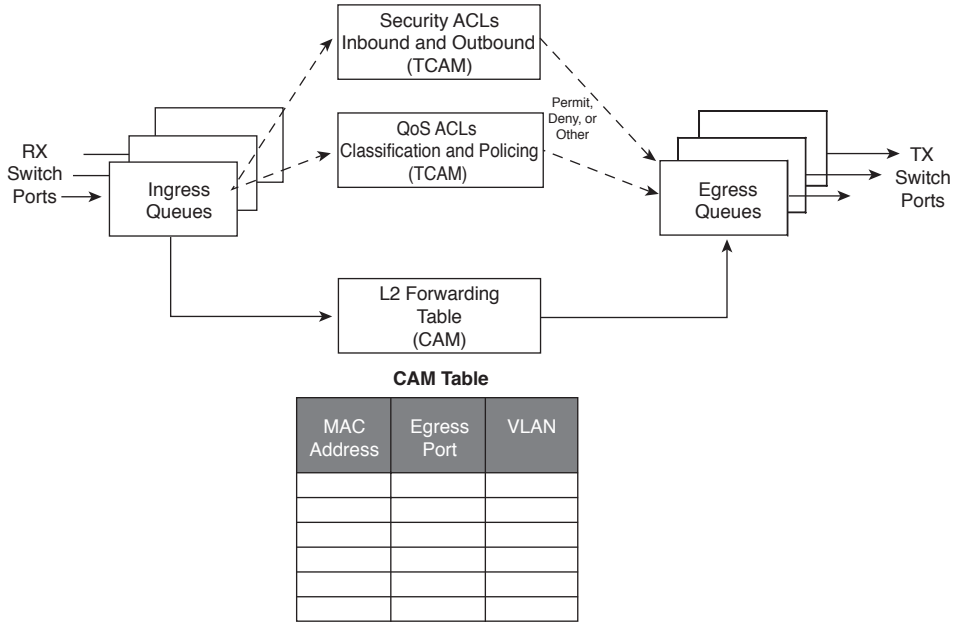


Figure 2-3 *Operations Within a Layer 2 Catalyst Switch*

When a frame arrives at a switch port, it is placed into one of the port's ingress queues. The queues each can contain frames to be forwarded, with each queue having a different priority or service level. The switch port then can be fine-tuned so that important frames get processed and forwarded before less-important frames. This can prevent time-critical data from being "lost in the shuffle" during a flurry of incoming traffic.

As the ingress queues are serviced and a frame is pulled off, the switch must figure out not only *where* to forward the frame, but also *whether* it should be forwarded and *how*. Three fundamental decisions must be made: one concerned with finding the egress switch port, and two concerned with forwarding policies. All these decisions are made *simultaneously* by independent portions of switching hardware and can be described as follows:

- **L2 forwarding table:** The frame's destination MAC address is used as an index, or key, into the content-addressable memory (CAM), or address, table. If the address is found, the egress switch port and the appropriate VLAN ID are read from the table. (If the address is not found, the frame is marked for flooding so that it is forwarded out every switch port in the VLAN.)
- **Security ACLs:** Access control lists (ACL) can be used to identify frames according to their MAC addresses, protocol types (for non-IP frames), IP addresses, protocols, and Layer 4 port numbers. The ternary content-addressable memory (TCAM) contains ACLs in a compiled form so that a decision can be made on whether to forward a frame in a single table lookup.
- **QoS ACLs:** Other ACLs can classify incoming frames according to quality of service (QoS) parameters, to police or control the rate of traffic flows, and to mark QoS parameters in outbound frames. The TCAM is also used to make these decisions in a single table lookup.

The CAM and TCAM tables are discussed in greater detail in the "Content-Addressable Memory" and "Ternary Content-Addressable Memory" sections, later in this chapter. After the CAM and TCAM table lookups have occurred, the frame is placed into the appropriate egress queue on the appropriate outbound switch port. The egress queue is determined by QoS values either contained in the frame or passed along with the frame. Like the ingress queues, the egress queues are serviced according to importance or time criticality; higher priority frames are sent out without being delayed by other outbound traffic.

Multilayer Switch Operation

Many Cisco Catalyst switches can also forward frames based on Layers 3 and 4 information contained in packets. This is known as *multilayer switching (MLS)*. Naturally, Layer 2 switching is performed at the same time because even the higher-layer encapsulations still are contained in Ethernet frames.

Types of Multilayer Switching

Catalyst switches have supported two basic generations or types of MLS: route caching (first-generation MLS) and topology-based (second-generation MLS). This section presents an overview of both, although only the second generation is supported in the Cisco IOS Software–based switch families, such as the Catalyst 2960, 3750, 4500, and 6500. You should understand the two types and the differences between them:

- **Route caching:** The first generation of MLS, requiring a route processor (RP) and a switch engine (SE). The RP must process a traffic flow's first packet to determine the destination. The SE listens to the first packet and to the resulting destination, then sets up a "shortcut" entry in its MLS cache. The SE forwards subsequent packets belonging to the same traffic flow based on shortcut entries in its cache.

  This type of MLS also is known by the names *Netflow LAN switching*, *flow-based* or *demand-based switching*, and "*route once, switch many*." The RP must examine each new traffic flow and set up shortcut entries for the SE. Even if this method isn't used to forward packets in Cisco IOS–based Catalyst switches, the technique can still be used to generate traffic flow information and statistics.

- **Topology-based:** The second generation of MLS, utilizing specialized hardware, is also organized with distinct RP and SE functions. The RP uses Layer 3 routing information to build and prepopulate a single database of the entire known network topology. This database becomes an efficient table lookup in hardware, and is consulted so that packets can be forwarded at high rates by the SE. The longest match found in the database is used as the correct Layer 3 destination. As the routing topology changes over time, the database contained in the hardware can be updated dynamically with no performance penalty.

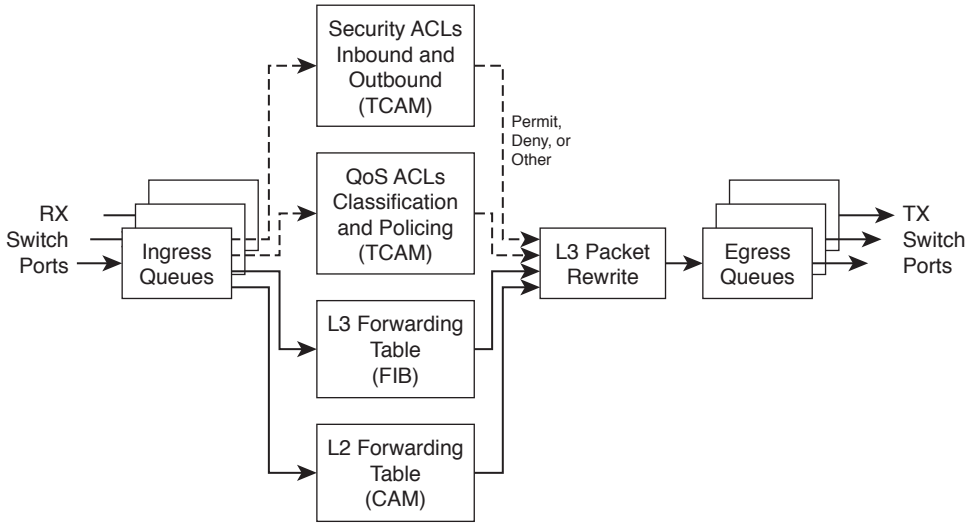
  This type of MLS is known as *Cisco Express Forwarding (CEF)*. A routing process running on the switch downloads the current routing table database into the *Forwarding Information Base (FIB)* area of hardware. CEF is discussed in greater detail in Chapter 12, "Multilayer Switching."

Tip Although the RP and SE functions within a multilayer switch do interact, they can operate independently, as if they are on different "planes." The control plane of a switch includes the RP and any process that runs to control or manage the switch, while the data plane exists in the SE, where data is forwarded.

Follow That Packet!

The path that a Layer 3 packet follows through a multilayer switch is similar to that of a Layer 2 switch. Obviously, some means of making a Layer 3 forwarding decision must be added. Beyond that, several, sometimes unexpected, things can happen to packets as they are forwarded. Figure 2-4 shows a typical multilayer switch and the decision processes

that must occur. Packets arriving on a switch port are placed in the appropriate ingress queue, just as in a Layer 2 switch.



CAM Table

MAC Address	Egress Port	VLAN

FIB Table

IP Address	Next-Hop IP Addr	Next-Hop MAC Addr	Egress Port

Figure 2-4 *Operations Within a Multilayer Catalyst Switch*

Each packet is pulled off an ingress queue and inspected for both Layer 2 and Layer 3 destination addresses. Now, the decision of *where* to forward the packet is based on two address tables, whereas the decision of *how* to forward the packet still is based on access list results.

All the multilayer switching decisions are performed simultaneously in hardware using the following functions:

- **L2 forwarding table:** The destination MAC address is used as an index into the CAM table. If the frame contains a Layer 3 packet that needs to be forwarded from one subnet to another, the destination MAC address will contain the address of a Layer 3 port on the switch itself. In this case, the CAM table results are used only to decide that the frame should be processed at Layer 3.

- **L3 forwarding table:** The FIB table is consulted, using the destination IP address as an index. The longest match in the table is found (both address and mask), and the resulting next-hop Layer 3 address is obtained. The FIB also contains each next-hop router's Layer 2 MAC address and the egress switch port (and VLAN ID) so that further table lookups are not necessary.
- **Security ACLs:** Inbound and outbound access lists are compiled into TCAM entries so that decisions of whether to forward a packet can be determined as a single table lookup.
- **QoS ACLs:** Packet classification, policing, and marking all can be performed as single table lookups in the QoS TCAM.

As with Layer 2 switching, the packet finally must be placed in the appropriate egress queue on the appropriate egress switch port.

During the multilayer switching process, some portions of the frame must be modified or rewritten, just as any router would do. For example, the destination MAC address in the inbound frame contains the address of the next-hop destination, which is the ingress Layer 3 interface on the multilayer switch. After the FIB table is consulted, the next-hop router IP and MAC addresses are found.

The next-hop Layer 2 address must be put into the frame in place of the original destination address (the multilayer switch). The frame's Layer 2 source address also must become that of the multilayer switch's egress interface before the frame is sent on to the next hop. As any good router must do, the Time-To-Live (TTL) value in the Layer 3 packet must be decremented by one.

Because the contents of the Layer 3 packet (the TTL value) have changed, the Layer 3 header checksum must be recalculated. And because both Layers 2 and 3 contents have changed, the Layer 2 checksum must be recalculated. In other words, the entire Ethernet frame must be rewritten before it goes into the egress queue. This also is accomplished efficiently in hardware.

Multilayer Switching Exceptions

To forward packets using the simultaneous decision processes described in the preceding section, the packet must be "MLS-ready" and must require no additional decisions. For example, CEF can directly forward most IP and IPv6 packets between hosts. This occurs when the source and destination addresses (both MAC and IP) are already known and no other IP parameters must be manipulated.

Other packets cannot be directly forwarded by CEF and must be handled in more detail. This is done by a quick inspection during the forwarding decisions. If a packet meets criteria such as the following, it is flagged for further processing and sent or "punted" to the switch CPU for *process switching*:

- ARP requests and replies
- IP packets requiring a response from a router (TTL has expired, MTU is exceeded, fragmentation is needed, and so on)

- IP broadcasts that will be relayed as unicast (DHCP requests, IP helper-address functions)
- Routing protocol updates
- Cisco Discovery Protocol packets
- Packets needing encryption
- Packets triggering Network Address Translation (NAT)
- Legacy multiprotocol packets (IPX, AppleTalk, and so on)

As you might expect, packets that are punted to the CPU cannot be forwarded as efficiently as packets that can be forwarded in hardware directly. The additional processing takes more time and consumes more CPU resources. Ideally, all packets should be forwarded in hardware, but that isn't always possible.

Tables Used in Switching

Catalyst switches maintain several types of tables to be used in the switching process. The tables are tailored for Layer 2 switching or MLS and are kept in very fast memory so that many fields within a frame or packet can be compared in parallel.

Content-Addressable Memory

All Catalyst switch models use a CAM table for Layer 2 switching. As frames arrive on switch ports, the source MAC addresses are learned and recorded in the CAM table. The port of arrival and the VLAN are also recorded in the table, along with a time stamp. If a MAC address learned on one switch port moves to a different port, the most recent MAC address, arrival port, and time stamp are recorded and the previous entry is deleted. If a MAC address is found already present in the table with the correct arrival port, only its time stamp is updated.

Switches generally have large CAM tables so that many addresses can be looked up for frame forwarding. However, there is not enough table space to hold every possible address on large networks. To manage the CAM table space, *stale entries* (addresses that have not been heard from for a period of time) are aged out. By default, idle CAM table entries are kept for 300 seconds (5 minutes) before they are deleted. You can change the default setting using the following configuration command:

```
Switch(config)# mac address-table aging-time seconds
```

By default, MAC addresses are learned dynamically from incoming frames. You also can configure static CAM table entries that contain MAC addresses that might not be learned otherwise. To do this, use the following configuration command:

```
Switch(config)# mac address-table static mac-address vlan vlan-id {drop | inter-
face
type mod/num}
```

Note You should be aware that there is a slight discrepancy in the CAM table command syntax. Until Catalyst IOS version 12.1(11)EA1, the syntax for CAM table commands used the keywords **mac-address-table**, connected by hyphens. In more recent Cisco IOS versions, the syntax has changed to use the keywords **mac address-table** (first hyphen omitted). The Catalyst 4500 and 6500 IOS Software are exceptions, however, and continue to use the **mac-address-table** keyword form. Many switch platforms support either syntax to ease the transition.

Exactly what happens when a host's MAC address is learned on one switch port, and then the host moves so that it appears on a different switch port? Ordinarily, the host's original CAM table entry would have to age out after 300 seconds, while its address was learned on the new port. To avoid having duplicate CAM table entries during that time, a switch purges any existing entries for a MAC address that has just been learned on a different switch port. This is a safe assumption because MAC addresses are unique, and a single host should never be seen on more than one switch port unless problems exist in the network. If a switch notices that a MAC address is being learned on alternating switch ports, it generates an error message that flags the MAC address as "flapping" between interfaces.

Ternary Content-Addressable Memory

In traditional routing, ACLs can match, filter, or control specific traffic. Access lists are made up of one or more access control entities (ACE) or matching statements that are evaluated in sequential order. Evaluating an access list can take up additional time, adding to the latency of forwarding packets.

In multilayer switches, however, all the matching processes that ACLs provide are implemented in hardware called a TCAM. With a TCAM, a packet can be evaluated against an entire access list within a single table lookup. Most switches have multiple TCAMs so that both inbound and outbound security and QoS ACLs can be evaluated simultaneously, or entirely in parallel with a Layer 2 or Layer 3 forwarding decision.

The Catalyst IOS Software has two components that are part of the TCAM operation:

- **Feature Manager (FM):** After an access list has been created or configured, the Feature Manager software compiles, or merges, the ACEs into entries in the TCAM table. The TCAM then can be consulted at full frame-forwarding speed.
- **Switching Database Manager (SDM):** On some Catalyst switche models, the TCAM is partitioned into several areas that support different functions. The SDM software configures or tunes the TCAM partitions, if needed, to provide ample space for specific switching functions. (The TCAM is fixed on Catalyst 4500 and 6500 platforms and cannot be repartitioned.)

TCAM Structure

The TCAM is an extension of the CAM table concept. Recall that a CAM table takes in an index or key value (usually a MAC address) and looks up the resulting value (usually a switch port or VLAN ID). Table lookup is fast and always based on an exact key match consisting of binary numbers made up of two possible values: 0 and 1 bits.

TCAM also uses a table-lookup operation but is greatly enhanced to allow a more abstract operation. For example, binary values (0s and 1s) make up a key into the table, but a mask value also is used to decide which bits of the key are actually relevant. This effectively makes a key consisting of three input values: 0, 1, and X (don't care) bit values—a threefold or *ternary* combination.

TCAM entries are composed of Value, Mask, and Result (VMR) combinations. Fields from frame or packet headers are fed into the TCAM, where they are matched against the value and mask pairs to yield a result. As a quick reference, these can be described as follows:

- **Values** are always 134-bit quantities, consisting of source and destination addresses and other relevant protocol information—all patterns to be matched. The information concatenated to form the value depends on the type of access list, as shown in Table 2-2. Values in the TCAM come directly from any address, port, or other protocol information given in an ACE, up to a maximum of 134 bits.

Table 2-2 *TCAM Value Pattern Components*

Access List Type	Value and Mask Components, (Number of Bits)
Ethernet	Source MAC (48), destination MAC (48), Ethertype (16)
ICMP	Source IP (32), destination IP (32), protocol (16), ICMP code (8), ICMP type (4), IP type of service (ToS) (8)
Extended IP using TCP/UDP	Source IP (32), destination IP (32), protocol (16), IP ToS (8), source port (16), source operator (4), destination port (16), destination operator (4)
Other IP	Source IP (32), destination IP (32), protocol (16), IP ToS (8)
IGMP	Source IP (32), destination IP (32), protocol (16), IP ToS (8), IGMP message type (8)

- **Masks** are also 134-bit quantities, in exactly the same format, or bit order, as the values. Masks select only the value bits of interest; a mask bit is set to mark a value bit to be exactly matched, or is not set to mark a value bit does not matter. The masks used in the TCAM stem from address or bit masks in ACEs.
- **Results** are numeric values that represent what action to take after the TCAM lookup occurs. Whereas traditional access lists offer only a permit or deny result, TCAM lookups offer a number of possible results or actions. For example, the result can be a permit or deny decision, an index value to a QoS policer, a pointer to a next-hop routing table, and so on.

Note This section discusses TCAM from an IPv4 perspective. When a dual IPv4-IPv6 SDM template is used, the TCAM becomes more limited in size. Because IPv6 addresses are 128 bits in length, some address compression must be used to store them in TCAM entries.

Key Topic

The TCAM is always organized by masks, where each unique mask has eight value patterns associated with it. For example, the Catalyst 6500 TCAM (one for security ACLs and one for QoS ACLs) holds up to 4096 masks and 32,768 value patterns. The trick is that each of the mask-value pairs is evaluated *simultaneously*, or in parallel, revealing the best or longest match in a single table lookup.

TCAM Example

Figure 2-5 shows how the TCAM is built and used. This is a simple example and might or might not be identical to the results that the Feature Manager produces because the ACEs might need to be optimized or rewritten to achieve certain TCAM algorithm requirements.

The sample access list 100 (extended IP) is configured and merged into TCAM entries. First, the mask values must be identified in the access list. When an address value and a corresponding address mask are specified in an ACE, those mask bits must be set for matching. All other mask bits can remain in the "don't care" state because they won't be used.

The access list contains only three unique masks: one that matches all 32 bits of the source IP address (found with an address mask of 0.0.0.0 or the keyword **host**), one that matches 16 bits of the destination address (found with an address mask of 0.0.255.255), and one that matches only 24 bits of the destination address (found with an address mask of 0.0.0.255). The keyword **any** in the ACEs means "match anything" or "don't care."

The three unique masks are placed into the TCAM. Then, for each mask, all possible value patterns are identified. For example, a 32-bit source IP mask (Mask 1) can be found only in ACEs with a source IP address of 192.168.199.14 and a destination of 10.41.0.0. (The rest of Mask 1 is the destination address mask 0.0.255.255.) Those address values are placed into the first value pattern slot associated with Mask 1. Mask 2 (0.0.255.255) has three value patterns: destination addresses 192.168.100.0, 192.168.5.0, and 192.168.199.0. Each of these is placed in the three pattern positions of Mask 2. This process continues until all ACEs have been merged.

When a mask's eighth pattern position has been filled, the next pattern with the same mask must be placed under a new mask in the table. A bit of a balancing act occurs to try to fit all ACEs into the available mask and pattern entries without an overflow.

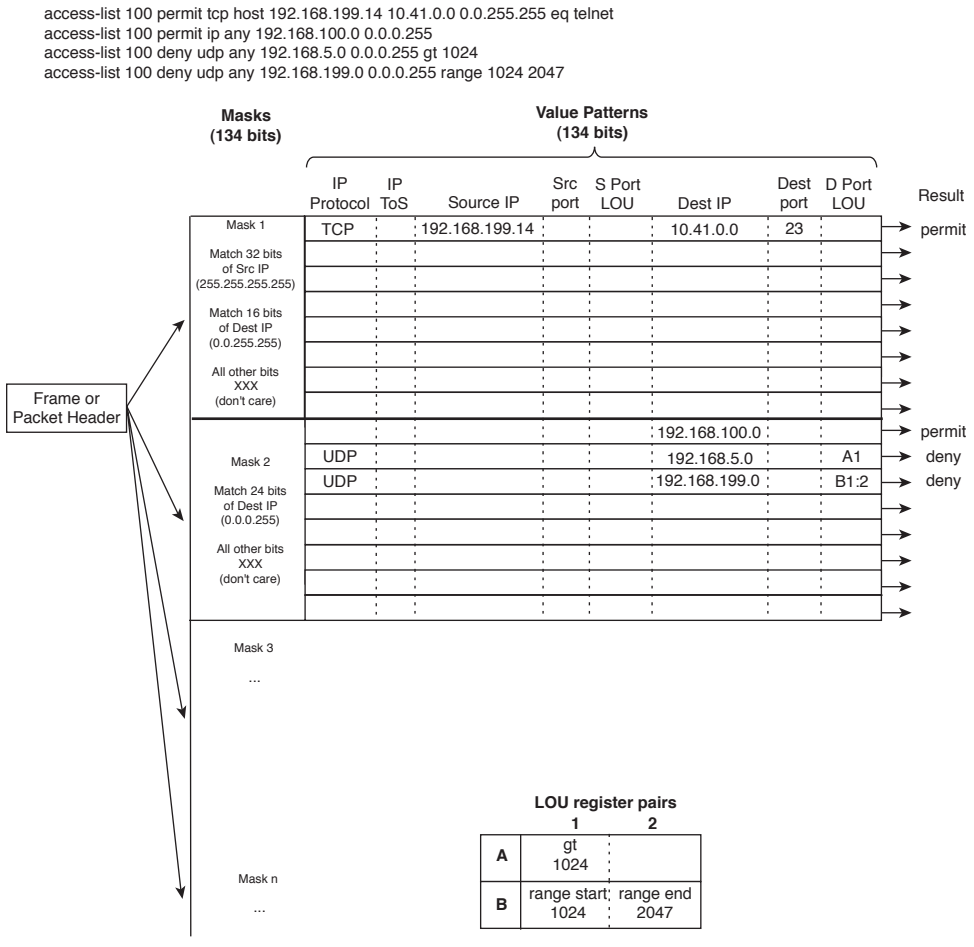


Figure 2-5 *How an Access List Is Merged into TCAM*

Port Operations in TCAM

You might have noticed that matching strictly based on values and masks covers only ACE statements that involve exact matches (either the eq port operation keyword or no Layer 4 port operations). For example, ACEs such as the following involve specific address values, address masks, and port numbers:

```
access-list test permit ip 192.168.254.0 0.0.0.255 any
access-list test permit tcp any host 192.168.199.10 eq www
```

What about ACEs that use port operators, where a comparison must be made? Consider the following:

```
access-list test permit udp any host 192.168.199.50 gt 1024
access-list test permit tcp any any range 2000 2002
```

A simple logical operation between a mask and a pattern cannot generate the desired result. The TCAM also provides a mechanism for performing a Layer 4 operation or comparison, also done during the single table lookup. If an ACE has a port operator, such as **gt, lt, neq**, or **range**, the Feature Manager software compiles the TCAM entry to include the use of the operator and the operand in a logical operation unit (LOU) register. Only a limited number of LOUs are available in the TCAM. If there are more ACEs with comparison operators than there are LOUs, the Feature Manager must break up the ACEs into multiple ACEs with only regular matching (using the eq operator).

In Figure 2-5, two ACEs require a Layer 4 operation:

- One that checks for UDP destination ports greater than 1024
- One that looks for the UDP destination port range 1024 to 2047

The Feature Manager checks all ACEs for Layer 4 operation and places these into LOU register pairs. These can be loaded with operations, independent of any other ACE parameters. The LOU contents can be reused if other ACEs need the same comparisons and values. After the LOUs are loaded, they are referenced in the TCAM entries that need them. This is shown by LOUs A1 and the B1:2 pair. A finite number (actually, a rather small number) of LOUs are available in the TCAM, so the Feature Manager software must use them carefully.

Managing Switching Tables

You can display or query the switching tables to verify the information that the switch has learned. As well, you might want to check the tables to find out on which switch port a specific MAC address has been learned. You can also manage the size of the various switching tables to optimize performance.

CAM Table Operation

To view the contents of the CAM table, you can use the following form of the **show mac address-table** EXEC command:

```
Switch# show mac address-table dynamic [address mac-address | interface type
  mod/num | vlan vlan-id]
```

The entries that have been learned dynamically will be shown. You can add the address keyword to specify a single MAC address, or the **interface** or **vlan** keyword to see addresses that have been learned on a specific interface or VLAN.

For example, assume that you need to find the learned location of the host with MAC address 0050.8b11.54da. The **show mac address-table dynamic address 0050.8b11.54da** command might produce the output in Example 2-1.

Example 2-1 *Determining Host Location by MAC Address*

```
Switch# show mac address-table dynamic address 0050.8b11.54da

          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type       Ports
----    -----------       ----       -----
  54    0050.8b11.54da    DYNAMIC    Gi1/0/1
Total Mac Addresses for this criterion: 1
Switch#
```

From this output, you can see that the host is somehow connected to interface Gigabit Ethernet 1/0/1, on VLAN 54.

Tip If your Catalyst IOS switch is not accepting commands of the form **mac address-table**, try adding a hyphen between the keywords. For example, the Catalyst 4500 and 6500 most likely will accept **show mac-address-table** instead.

Suppose that this same command produced no output, showing nothing about the interface and VLAN where the MAC address is found. What might that mean? Either the host has not sent a frame that the switch can use for learning its location, or something odd is going on. Perhaps the host is using two network interface cards (NIC) to load balance traffic; one NIC is only receiving traffic, whereas the other is only sending. Therefore, the switch never hears and learns the receiving-only NIC address.

To see all the MAC addresses that are currently found on interface GigabitEthernet1/0/29, you could use the **show mac address-table dynamic interface gig1/0/29** command. The output shown in Example 2-2 indicates that only one host has been learned on the interface. Perhaps only a single PC connects to that interface.

Example 2-2 *Determining Hosts Active on an Interface*

```
Switch# show mac address-table dynamic interface gigabitethernet1/0/29

          Mac Address Table
-----------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       ----        -----
 537    0013.7297.3d4b    DYNAMIC     Gi1/0/29
Total Mac Addresses for this criterion: 1
Switch#
```

However, suppose the same command is used to check interface GigabitEthernet1/1/1. The output shown in Example 2-3 lists many MAC addresses—all found on a single interface. How can so many addresses be learned on one switch interface? This interface must lead to another switch or another part of the network where other devices are located. As frames have been received on GigabitEthernet1/1/1, coming from the other devices, the local switch has added the source MAC addresses into its CAM table.

Example 2-3 *Finding Many Hosts on an Interface*

```
Switch# show mac address-table dynamic interface gig1/1/1

          Mac Address Table
-----------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       ----        -----
 580    0000.0c07.ac01    DYNAMIC     Gi1/1/1
 580    0007.0e0b.f918    DYNAMIC     Gi1/1/1
 580    000f.1f78.1094    DYNAMIC     Gi1/1/1
 580    0011.43ac.b083    DYNAMIC     Gi1/1/1
 580    0011.bb2d.3f6e    DYNAMIC     Gi1/1/1
 580    0014.6a86.1f1e    DYNAMIC     Gi1/1/1
 580    0014.6a86.1f3d    DYNAMIC     Gi1/1/1
 580    0014.6a86.1f3f    DYNAMIC     Gi1/1/1
 580    0014.6a86.1f47    DYNAMIC     Gi1/1/1
 —More—
```

Tip Frequently, you need to know where a user with a certain MAC address is connected. In a large network, discerning at which switch and switch port a MAC address can be found might be difficult. Start at the network's center, or core, and display the CAM table entry for the user's MAC address. Look at the switch port shown in the entry and find the neighboring switch connected to that port using CDP neighbor information. Then move to that switch and repeat the CAM table query process. Keep moving from switch to switch until you reach the edge of the network where the MAC address physically connects.

To see the CAM table's size, use the **show mac address-table count** command, as shown in Example 2-4. MAC address totals are shown for each active VLAN on the switch, as well as the total number of spaces remaining in the CAM table. This can give you a good idea of the size of the CAM table and how many hosts are using the network.

Example 2-4 *Checking the Size of the CAM Table*

```
Switch# show mac address-table count

Mac Entries for Vlan 1:
---------------------------
Dynamic Address Count  : 0
Static  Address Count  : 0
Total Mac Addresses    : 0

Mac Entries for Vlan 2:
---------------------------
Dynamic Address Count  : 89
Static  Address Count  : 0
Total Mac Addresses    : 89

Mac Entries for Vlan 580:
----------------------------
Dynamic Address Count  : 244
Static  Address Count  : 0
Total Mac Addresses    : 244

Total Mac Address Space Available: 5791
Switch#
```

CAM table entries can be cleared manually, if needed, by using the following EXEC command:

```
Switch# clear mac address-table dynamic [address mac-address | interface type
  mod/num | vlan vlan-id]
```

TCAM Operation

The TCAM in a switch is more or less self-sufficient. Access lists are compiled or merged automatically into the TCAM, so there is nothing to configure. The only concept you need to be aware of is how the TCAM resources are being used. You can use the **show platform tcam utilization** EXEC command shown in Example 2-5 to get an idea of the TCAM utilization. Compare the Used Masks/Values entries to the Max Masks/Values entries.

Example 2-5 *Displaying TCAM Utilization*

```
Switch#show platform tcam utilization

CAM Utilization for ASIC# 0                    Max            Used
                                             Masks/Values    Masks/Values

 Unicast mac addresses:                       6364/6364        311/311
 IPv4 IGMP groups + multicast routes:         1120/1120          8/8
 IPv4 unicast directly-connected routes:      6144/6144          0/0
 IPv4 unicast indirectly-connected routes:    2048/2048         28/28
 IPv4 policy based routing aces:               452/452          12/12
 IPv4 qos aces:                                512/512          21/21
 IPv4 security aces:                           964/964          33/33

Note: Allocation of TCAM entries per feature uses
a complex algorithm. The above information is meant
to provide an abstract view of the current TCAM utilization
Switch#
```

TCAMs have a limited number of usable mask, value pattern, and LOU entries. If access lists grow to be large or many Layer 4 operations are needed, the TCAM tables and registers can overflow. If that happens while you are configuring an ACL, the switch will generate syslog messages that flag the TCAM overflow situation as it tries to compile the ACL into TCAM entries.

Managing Switching Table Sizes

High-end Cisco switches are designed for efficient multilayer switching at any location within a network. For example, the versatile Catalyst 4500 and 6500 models can be used equally well in the core, distribution, or access layer because their hardware contains ample switching engines and table space for any application. Other models such as the 2960, 3750, and 3850 have a fixed architecture with limited switching table space. The CAM, FIB, and other tables must all share resources; for one table to grow larger, the others must shrink.

Fortunately, you can select a preferred type of switching that in turn affects the relative size of the switching tables. To excel at Layer 2 switching, the CAM table should increase in size, while the FIB or routing table space should decrease. If a switch is used to route traffic, its FIB table space should grow and its CAM table should shrink.

The SDM manages the memory partitions in a switch. You can display the current partition preference and a breakdown of table sizes with the following EXEC command:

```
Switch#show sdm prefer
```

Example 2-6 shows that the switch is operating with the "desktop default" memory template, which is tailored for the access layer. According to the numbers, the desktop default template provides a balanced mix of Layer 2 (unicast MAC addresses, or the CAM table) and Layer 3 (IPv4 unicast routes, or the FIB table), as well as IPv4 access control lists, and some minimal support for IPv6.

Example 2-6 *Displaying the Current SDM Template*

```
Switch#show sdm prefer

 The current template is "desktop default" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 8 routed interfaces and 1024 VLANs.

  number of unicast mac addresses:                  6K
  number of IPv4 IGMP groups + multicast routes:    1K
  number of IPv4 unicast routes:                    8K
    number of directly-connected IPv4 hosts:        6K
    number of indirect IPv4 routes:                 2K
  number of IPv6 multicast groups:                  64
  number of directly-connected IPv6 addresses:      74
  number of indirect IPv6 unicast routes:           32
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 0.875k
  number of IPv6 policy based routing aces:         0
  number of IPv6 qos aces:                          0
  number of IPv6 security aces:                     60
Switch#
```

You can configure a switch to operate based on other SDM templates by using the following global configuration command:

```
Switch(config)# sdm prefer template
```

The switch must then be rebooted for the new template to take effect. Tables 2-3 and 2-4 list the template types along with the number of entries allowed in each memory partition. The two shaded rows represent the CAM and FIB table spaces. To get a feel for the SDM templates, notice which function is favored in each of the template types. The Unicast MAC Addresses and Unicast Routes rows are highlighted as examples.

Don't worry about memorizing the tables and their contents; rather, you should know how to display the current template and how to configure a new one.

Table 2-3 *IPv4 SDM Templates and Memory Partitions*

Memory Partition	SDM Template Type Keyword			
	default	access	vlan	routing
Unicast MAC Addresses	6 K	4 K	12 K	3 K
IPv4 IGMP Groups + Multicast Routes	1 K	1 K	1 K	1 K
IPv4 Unicast Routes	8 K	6 K	0	11 K
Directly Connected IPv4 Hosts	6 K	4 K	0	3 K
Indirect IPv4 Routes	2 K	2 K	0	8 K
IPv4 Policy Based Routing ACEs	0	0.5 K	0	0.5 K
IPv4/MAC QoS ACEs	0.5 K	0.5 K	0.5 K	0.375 K
IPv4/MAC Security ACEs	1 K	2 K	1 K	1 K
VLANs	1 K	1 K	1 K	1 K

Table 2-4 *Dual IPv4-IPv6 SDM Templates and Memory Partitions*

Memory Partition	SDM Template Type Keyword			
	dual-ipv4-and-ipv6			indirect-ipv4-and-ipv6
	default	vlan	routing	
Unicast MAC Addresses	2 K	8 K	1.5 K	2 K
IPv4 IGMP Groups + Multicast Routes	1 K	1 K IGMP0 mmulticast	1 K	1 K
IPv4 Unicast Routes	3 K	0	2.7 K	4 K
Directly Connected IPv4 Hosts	2 K	0	1.5 K	2 K
Indirect IPv4 Routes	1 K	0	1.2 K	2 K
IPv6 Multicast Groups	1 K	1 K	1 K	1 K
Directly Connected IPv6 Addresses	2 K	0	1.5 K	2 K
Indirect IPv6 Unicast Routes	1 K	0.125 K	1.25 K	3 K
IPv4 Policy Based Routing ACEs	0	0	0.25 K	0.125 K
IPv4/MAC QoS ACEs	0.5 K	0.5 K	0.5 K	0.5 K
IPv4/MAC Security ACEs	1 K	1 K	0..5 K	0.625 K
IPv6 Policy Based Routing ACEs	0	0	0.25 K	0.125 K
IPv6 QoS ACEs	0.5 K	0.5 K	0.5 K	0.125 K
IPv6 Security ACEs	0.5 K	0.5 K	0.5 K	0.125 K

Exam Preparation Tasks

Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the outer margin of the page. Table 2-5 lists a reference of these key topics and the page numbers on which each is found.

Key Topic

Table 2-5 *Key Topics for Chapter 2*

Key Topic Element	Description	Page Number
Paragraph	Discusses collision domain	6
Paragraph	Discusses flooding and unknown unicast flooding	8
List	Describes topology-based switching	11
Paragraph	Discusses the CAM table	14
Paragraph	Explains TCAM operation	17

Define Key Terms

Define the following key terms from this chapter, and check your answers in the glossary:

collision domain, flooding, unknown unicast flooding, CEF, FIB, CAM, TCAM, SDM

Use Command Reference to Check Your Memory

This section includes the most important configuration and EXEC commands covered in this chapter. It might not be necessary to memorize the complete syntax of every command, but you should remember the basic keywords that are needed.

To test your memory of the CAM-related commands, cover the right side of Table 2-6 with a piece of paper, read the description on the left side, and then see how much of the command you can remember.

Remember that the CCNP exam focuses on practical or hands-on skills that are used by a networking professional. For most of the skills covered in this chapter, remember that the commands always involve the keywords **mac address-table**.

Table 2-6 *Commands Used to Monitor and Manipulate the CAM Table*

Function	Command
Find the location of a specific MAC address.	**show mac address-table dynamic address** *mac-address*
Display all MAC addresses learned on a specific interface.	**show mac address-table dynamic interface** *type number*
Display the current CAM table size.	**show mac address-table count**
Enter a static CAM table entry.	**mac address-table static** *mac-address* **vlan** *vlan-id* {**drop** | **interface** *type number*}
Clear a CAM entry.	**clear mac address-table dynamic** [**address** *mac-address* | **interface** *type number* | **vlan** *vlan-id*]
Display TCAM utilization.	**show platform tcam utilization**
Display the current memory template.	**show sdm prefer**
Configure a preferred memory template.	**sdm prefer** *template*

Official Cert Guide

Learn, prepare, and practice for exam success

CCNP

Routing and Switching TSHOOT 300-135

RAYMOND LACOSTE
KEVIN WALLACE, CCIE® No. 7945

ciscopress.com

Contents

This chapter covers the following topics:

- **Troubleshooting Switch Performance Issues:** This section identifies common reasons why a switch might not be performing as expected.
- **Troubleshooting Router Performance Issues:** This section identifies common reasons why a router might not be performing as expected.

CHAPTER 3

Troubleshooting Device Performance

Switches and routers are comprised of many different components. For example, they contain a processor, memory (volatile such as RAM and nonvolatile such as NVRAM and flash), and various interfaces. They are also responsible for performing many different tasks, such as routing, switching, and building all the necessary tables and structures needed to perform various tasks.

The building of the tables and structures is done by the CPU. The storage of these tables and structures is in some form of memory. The routers and switches forward traffic from one interface to another interface based on these tables and structures. Therefore, if a router's or switch's CPU is constantly experiencing high utilization, the memory is overloaded, or the interface buffers are full, these devices will experience performance issues.

In this chapter, we will discuss common reasons for high CPU and memory utilization on routers and switches in addition to how we can recognize them. We will also examine interface statistics as they can be an initial indication of some type of issue.

"Do I Know This Already?" Quiz

The "Do I Know This Already?" quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the "Exam Preparation Tasks" section. If you are in doubt about your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. Table 3-1 lists the major headings in this chapter and their corresponding "Do I Know This Already?" quiz questions. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes."

Table 3-1 *Do I Know This Already?" Section-to-Question Mapping*

Foundation Topics Section	Questions
Troubleshooting Switch Performance Issues	1–4
Troubleshooting Router Performance Issues	5–8

Caution The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, you should mark that question as wrong for purposes of the self-assessment. Giving yourself credit for an answer that you correctly guess skews your self-assessment results and might provide you with a false sense of security.

1. What are the components of a switch's control plane? (Choose two.)
   - **a.** Backplane
   - **b.** Memory
   - **c.** CPU
   - **d.** Forwarding logic
2. What are good indications that you have a duplex mismatch? (Choose two.)
   - **a.** The half-duplex side of the connection has a high number of FCS errors.
   - **b.** The full-duplex side of the connection has a high number of FCS errors.
   - **c.** The half-duplex side of the connection has a high number of late collisions.
   - **d.** The full-duplex side of the connection has a high number of late collisions.
3. Which of the following are situations when a switch's TCAM would punt a packet to the switch's CPU? (Choose the three best answers.)
   - **a.** OSPF sends a multicast routing update.
   - **b.** An administrator telnets to a switch.
   - **c.** An ACL is applied to a switch port.
   - **d.** A switch's TCAM has reached capacity.
4. The output of a **show processes cpu** command on a switch displays the following in the first line of the output:

   ```
   CPU utilization for five seconds: 10%/7%; one minute: 12%; five
   minutes: 6%
   ```

   Based on the output, what percent of the switch's CPU is being consumed with interrupts?
   - **a.** 10 percent
   - **b.** 7 percent
   - **c.** 12 percent
   - **d.** 6 percent

5. Which router process is in charge of handling interface state changes?

   a. TCP Timer process

   b. IP Background process

   c. Net Background process

   d. ARP Input process

6. Which of the following is the least efficient (that is, the most CPU intensive) of a router's packet-switching modes?

   a. Fast switching

   b. CEF

   c. Optimum switching

   d. Process switching

7. What command is used to display the contents of a routers FIB?

   a. **show ip cache**

   b. **show processes cpu**

   c. **show ip route**

   d. **show ip cef**

8. Identify common reasons that a router displays a MALLOCFAIL error. (Choose the two best answers.)

   a. Cisco IOS bug

   b. Security issue

   c. QoS issue

   d. BGP filtering

Foundation Topics

Troubleshooting Switch Performance Issues

Switch performance issues can be tricky to troubleshoot because the problem reported is often subjective. For example, if a user reports that the network is running "slowly," the user's perception might mean that the network is slow compared to what he expects. However, network performance might very well be operating at a level that is hampering productivity and at a level that is indeed below its normal level of operation. At that point, as part of the troubleshooting process, you need to determine what network component is responsible for the poor performance. Rather than a switch or a router, the user's client, server, or application could be the cause of the performance issue.

If you do determine that the network performance is not meeting technical expectations (as opposed to user expectations), you should isolate the source of the problem and diagnose the problem on that device. This section assumes that you have isolated the device causing the performance issue, and that device is a Cisco Catalyst switch.

Cisco Catalyst Switch Troubleshooting Targets

Cisco offers a variety of Catalyst switch platforms, with different port densities, different levels of performance, and different hardware. Therefore, troubleshooting switches will be platform dependent. Many similarities do exist, however. For example, all Cisco Catalyst switches include the following components:

Key Topic

- **Ports:** A switch's ports physically connect the switch to other network devices. These ports (also known as *interfaces)* allow a switch to receive and transmit traffic.
- **Forwarding logic:** A switch contains hardware that makes forwarding decisions based on different tables in the data plane.
- **Backplane:** A switch's backplane physically interconnects a switch's ports. Therefore, depending on the specific switch architecture, frames flowing through a switch enter through a port (that is, the ingress port), flow across the switch's backplane, and are forwarded out of another port (that is, an egress port).
- **Control plane:** A switch's CPU and memory reside in the control plane. This control plane is responsible for running the switch's operating system and building the necessary structures used to make forwarding decisions—for example, the MAC address table and the spanning-tree topology to name a few.

Figure 3-1 depicts these components within a switch. Notice that the control plane does not directly participate in the frame-forwarding process. However, the forwarding logic contained in the forwarding hardware comes from the control plane. Therefore, there is an indirect relationship between frame forwarding and the control plane. As a result, a continuous load on the control plane could, over time, impact the rate at which the switch forwards frames. Also, if the forwarding hardware is operating at maximum

capacity, the control plane begins to provide the forwarding logic. So, although the control plane does not architecturally appear to impact switch performance, it should be considered when troubleshooting.

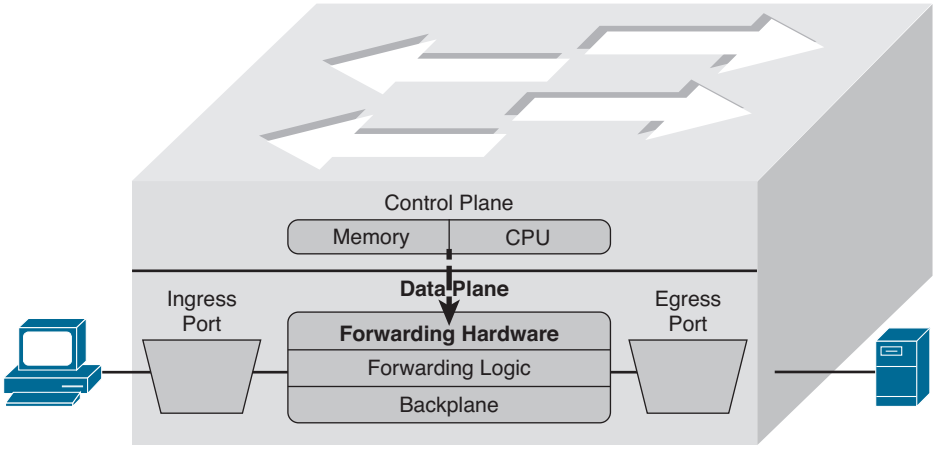


Figure 3-1 *Cisco Catalyst Switch Hardware Components*

The following are two common troubleshooting targets to consider when diagnosing a suspected switch issue:

- Port errors
- Mismatched duplex settings

The sections that follow evaluate these target areas in greater detail.

Port Errors

When troubleshooting a suspected Cisco Catalyst switch issue, a good first step is to check port statistics. For example, examining port statistics can let a troubleshooter know whether an excessive number of frames are being dropped. If a TCP application is running slowly, the reason might be that TCP flows are going into *TCP* slow start, which causes the window size, and therefore the bandwidth efficiency, of TCP flows to be reduced. A common reason that a TCP flow enters slow start is packet drops. Similarly, packet drops for a UDP flow used for voice or video could result in noticeable quality degradation, because dropped UDP segments are not retransmitted.

Although dropped frames are most often attributed to network congestion, another possibility is that the cabling could be bad. To check port statistics, a troubleshooter could leverage the **show interfaces** command. Consider Example 3-1, which shows the output of the **show interfaces gig 0/9 counters** command on a Cisco Catalyst 3750-E switch. Notice that this output shows the number of inbound and outbound frames seen on the specified port.

Example 3-1 ***show interfaces gig 0/9 counters*** *Command Output*

```
SW1# show interfaces gig 0/9 counters
Port            InOctets     InUcastPkts     InMcastPkts     InBcastPkts
Gi0/9           31265148           20003            3179               1

Port           OutOctets    OutUcastPkts    OutMcastPkts    OutBcastPkts
Gi0/9           18744149           9126              96               6
```

To view errors that occurred on a port, you could add the keyword of errors after the **show interfaces** *interface_id* **counters** command. Example 3-2 illustrates sample output from the **show interfaces gig 0/9 counters errors** command.

Example 3-2 ***show interfaces gig 0/9 counters errors*** *Command Output*

```
SW1# show interfaces gig 0/9 counters errors
Port        Align-Err    FCS-Err   Xmit-Err    Rcv-Err UnderSize
Gi0/9               0          0          0          0         0

Port      Single-Col Multi-Col  Late-Col Excess-Col Carri-Sen     Runts    Giants
Gi0/9           5603          0       5373          0          0          0          0
```

Table 3-2 provides a reference for the specific errors that might show up in the output of the **show interfaces** [*interface_id*] **counters errors** command.

Table 3-2 *Errors in the show interfaces interface_id counters errors Command*

Key Topic

Error Counter	Description
Align-Err	An alignment error occurs when frames do not end with an even number of octets, while simultaneously having a bad Cyclic Redundancy Check (CRC). An alignment error normally suggests a Layer 1 issue, such as cabling or port (either switch port or NIC port) issues.
FCS-Err	A Frame Check Sequence (FCS) error occurs when a frame has an invalid checksum, although the frame has no framing errors. Like the Align-Err error, an FCS-Err often points to a Layer 1 issue, but it also occurs when there is a duplex mismatch.
Xmit-Err	A transmit error (that is, Xmit-Err) occurs when a port's transmit buffer overflows. A speed mismatch between inbound and outbound links often results in a transmit error.
Rcv-Err	A receive error (that is, Rcv-Err) occurs when a port's receive buffer overflows. Congestion on a switch's backplane could cause the receive buffer on a port to fill to capacity, as frames await access to the switch's backplane. However, most likely, a Rcv-Err is indicating a duplex mismatch.

Error Counter	Description
UnderSize	An undersize frame is a frame with a valid checksum but a size less than 64 bytes. This issue suggests that a connected host is sourcing invalid frame sizes.
Single-Col	A Single-Col error occurs when a single collision occurs before a port successfully transmits a frame. High bandwidth utilization on an attached link or a duplex mismatch are common reasons for a Single-Col error.
Multi-Col	A Multi-Col error occurs when more than one collision occurs before a port successfully transmits a frame. Similar to the Single-Col error, high bandwidth utilization on an attached link or a duplex mismatch are common reasons for a Multi-Col error.
Late-Col	A late collision is a collision that is not detected until well after the frame has begun to be forwarded. While a Late-Col error could indicate that the connected cable is too long, this is an extremely common error seen in mismatched duplex conditions.
Excess-Col	The Excess-Col error occurs when a frame experienced 16 successive collisions, after which the frame was dropped. This error could result from high bandwidth utilization, a duplex mismatch, or too many devices on a segment.
Carri-Sen	The Carri-Sen counter is incremented when a port wants to send data on a half-duplex link. This is normal and expected on a half-duplex port, because the port is checking the wire to make sure that no traffic is present prior to sending a frame. This operation is the carrier sense procedure described by the Carrier Sense Multiple Access with Collision Detect (CSMA/CD) operation used on half-duplex connections. Full-duplex connections, however, do not use CSMA/CD.
Runts	A runt is a frame that is less than 64 bytes in size and has a bad CRC. A runt could result from a duplex mismatch or a Layer 1 issue.
Giants	A giant is a frame size greater than 1518 bytes (assuming that the frame is not a jumbo frame) that has a bad FCS. Typically, a giant is caused by a problem with the NIC in an attached host. The jumbo frame has a frame size greater than 1518 bytes but it has a valid FCS.

Mismatched Duplex Settings

As seen in Table 3-2, duplex mismatches can cause a wide variety of port errors. Keep in mind that almost all network devices, other than shared media hubs, can run in full-duplex mode. Therefore, if you have no hubs in your network, all devices should be running in full-duplex mode.

Cisco Catalyst switch ports should be configured to autonegotiate both speed and duplex, which is the default setting. Two justifications for this recommendation are as follows:

- If a connected device only supported half-duplex, it would be better for a switch port to negotiate down to half-duplex and run properly than to be forced to run full-duplex, which would result in multiple errors.
- The automatic medium-dependent interface crossover (auto-MDIX) feature can automatically detect whether a port needs a crossover or a straight-through cable to interconnect with an attached device and adjust the port to work regardless of which cable type is connected. You can enable this feature in interface configuration mode with the **mdix auto** command on some models of Cisco Catalyst switches. However, the auto-MDIX feature requires that the port autonegotiate both speed and duplex.

In a mismatched duplex configuration, a switch port at one end of a connection is configured for full-duplex, whereas a switch port at the other end of a connection is configured for half-duplex. Among the different errors previously listed in Table 3-2, two of the biggest indicators of a duplex mismatch are a high FCS-Err counter or a high Late-Col counter. Specifically, a high FCS-Err counter is common to find on the full-duplex end of a connection with a mismatched duplex, while a high Late-Col counter is common on the half-duplex end of the connection.

To illustrate, examine Examples 3-3 and 3-4, which display output based on the topology depicted in Figure 3-2. Example 3-3 shows the half-duplex end of a connection, and Example 3-4 shows the full-duplex end of a connection. The half-duplex end sends a frame because it thinks it is safe to send based on the CSMA/CD rule. The full-duplex end sends a frame because it is always safe to send and a collision should not occur. When the collision occurs in this example, SW1 will cease to transmit the remainder of the frame because the port is half-duplex and record that there was a late collision. However, SW2 will continue to send and receive frames. The frames it receives will not be complete because SW1 did not send the entire frame. Therefore, the FCS (mathematical checksum) of the frame does not match and we have FCS errors on the full-duplex side.

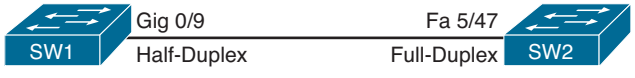


Figure 3-2 *Topology with Duplex Mismatch*

Example 3-3 *Output from the* ***show interfaces gig 0/9 counters errors*** *and the* ***show interfaces gig 0/9 | include duplex*** *Commands on a Half-Duplex Port*

```
SW1# show interfaces gig 0/9 counters errors
Port        Align-Err    FCS-Err   Xmit-Err    Rcv-Err UnderSize
Gi0/9               0          0          0          0         0

Port      Single-Col Multi-Col  Late-Col Excess-Col Carri-Sen   Runs   Giants
Gi0/9           5603          0       5373          0          0      0        0
```

```
SW1# show interfaces gig 0/9  include duplex
  Half-duplex, 100Mb/s, link type is auto, media type is 10/100/1000BaseTX
SW1#
```

Example 3-4 *Output from the* ***show interfaces fa 5/47 counters errors*** *and the* ***show interfaces fa 5/47 | include duplex*** *Commands on a Full-Duplex Port*

```
SW2# show interfaces fa 5/47 counters errors
v
Port        Align-Err    FCS-Err   Xmit-Err    Rcv-Err UnderSize OutDiscards
Fa5/47              0       5248          0       5603         27           0

Port   Single-Col Multi-Col  Late-Col Excess-Col Carri-Sen     Runs    Giants
Fa5/47          0          0          0          0          0       227         0

Port        SQETest-Err Deferred-Tx IntMacTx-Err IntMacRx-Err Symbol-Err
Fa5/47                0           0            0            0          0

SW2# show interfaces fa 5/47  include duplex
  Full-duplex, 100Mb/s
SW2#
```

In your troubleshooting, even if you only have access to one of the switches, if you suspect a duplex mismatch, you could change the duplex settings on the switch over which you do have control. Then, you could clear the interface counters to see whether the errors continue to increment. You could also perform the same activity (for example, performing a file transfer) that the user was performing when he noticed the performance issue. By comparing the current performance to the performance experienced by the user, you might be able to conclude that the problem has been resolved by correcting a mismatched duplex configuration.

TCAM Troubleshooting

As previously mentioned, the two primary components of forwarding hardware are forwarding logic and backplane. A switch's backplane, however, is rarely the cause of a switch performance issue, because most Cisco Catalyst switches have high-capacity backplanes. However, it is conceivable that in a modular switch chassis, the backplane will not have the throughput to support a fully populated chassis, where each card in the chassis supports the highest combination of port densities and port speeds.

The architecture of some switches allows groups of switch ports to be handled by separated hardware. Therefore, you might experience a performance gain by simply moving a cable from one switch port to another. However, to strategically take advantage of this design characteristic, you must be very familiar with the architecture of the switch with which you are working.

A multilayer switch's forwarding logic can impact switch performance. A switch's forwarding logic is compiled into a special type of memory called ternary content

addressable memory (TCAM), as illustrated in Figure 3-3. TCAM works with a switch's Cisco Express Forwarding (CEF) feature in the data plane (hardware) to provide extremely fast forwarding decisions. This is accomplished because information from the control plane relating to routing processes such as unicast routing, multicast routing, and policy-based routing, as well as information related to traffic policies such as security and QoS ACLs, is populated into the TCAM tables at the data plane (hardware). However, if a switch's TCAM is unable to forward traffic (for example, the TCAM table is full and does not have the information needed to forward the traffic), that traffic is sent (punted) to the CPU so it can be forwarded by the switch's CPU, which has a limited forwarding capability.

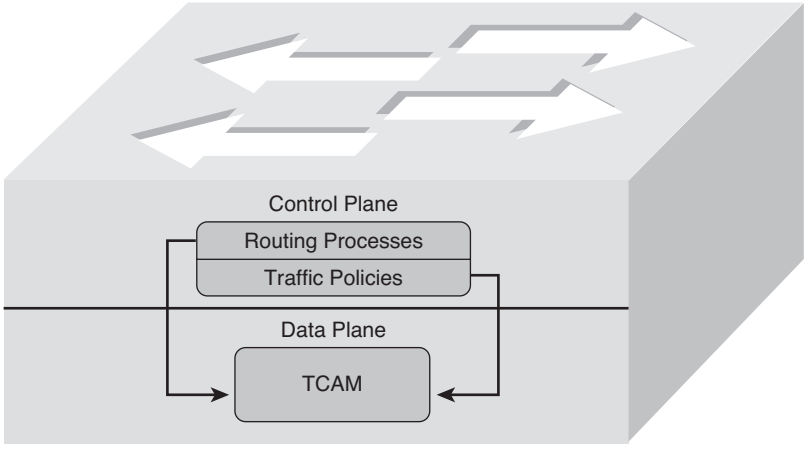


Figure 3-3 *Populating the TCAM*

The process of the TCAM sending packets to a switch's CPU is called *punting*. Consider a few reasons why a packet might be punted from a TCAM to its CPU:

Key Topic

- Routing protocols, in addition to other control plane protocols such as STP, that send multicast or broadcast traffic will have that traffic sent to the CPU for processing.
- Someone connecting to a switch administratively (for example, establishing a Telnet or SSH session with the switch) will have his packets sent to the CPU for processing.
- Packets using a feature not supported in hardware (for example, packets traveling over a GRE tunnel) are sent to the CPU for processing.
- If a switch's TCAM has reached capacity, additional packets will be punted to the CPU. A TCAM might reach capacity if it has too many installed routes or configured access control lists. This is usually the case when you attempt to use a lower-end switch in place of a higher-end switch to save money. This is not generally a good practice.

From the events listed, the event most likely to cause a switch performance issue is a TCAM filling to capacity. Therefore, when troubleshooting switch performance, you

might want to investigate the state of the switch's TCAM. Please be sure to check the documentation for your switch model, because TCAM verification commands can vary among platforms.

On most switch platforms, TCAMs cannot be upgraded. Therefore, if you conclude that a switch's TCAM is the source of the performance problems being reported, you could either use a switch with higher-capacity TCAMs or reduce the number of entries in a switch's TCAM. For example, you could try to optimize your access control lists by being more creative with the entries or leverage route summarization to reduce the number of route entries maintained by a switch's TCAM. Also, some switches (for example, Cisco Catalyst 2960, 3560, or 3750 Series switches) enable you to change the amount of TCAM memory allocated to different switch features. This allows you to "borrow" TCAM memory that was reserved for one feature and use it for another feature, optimizing the resources on the switch. This can be accomplished by changing the Switch Database Management (SDM) template on the switch. Refer to Example 3-5, which displays the TCAM resource utilization on a Catalyst 3750E switch. Notice how there is a finite amount of resources that have been reserved for various services and features on the switch. There is a maximum value for unicast MAC addresses, IPv4 unicast and multicast routes, as well as QoS and security access control entries. It appears from this example that SW2 has maxed out the amount of resources that are reserved for IPv4 unicast indirectly connected routes. Therefore, if a packet needs to be forwarded and the needed information is not in the TCAM, it will be punted to the CPU.

Example 3-5 ***show platform tcam utilization*** *Command Output on a Cisco Catalyst Switch*

```
SW2# show platform tcam utilization

CAM Utilization for ASIC# 0                     Max            Used
                                              Masks/Values    Masks/values

 Unicast mac addresses:                        6364/6364         35/35
 IPv4 IGMP groups + multicast routes:          1120/1120          1/1
 IPv4 unicast directly-connected routes:       6144/6144          9/9
 IPv4 unicast indirectly-connected routes:     2048/2048        2048/2048
 IPv4 policy based routing aces:                442/442          12/12
 IPv4 qos aces:                                 512/512          21/21
 IPv4 security aces:                            954/954          42/42

Note: Allocation of TCAM entries per feature uses
a complex algorithm. The above information is meant
to provide an abstract view of the current TCAM utilization
```

To reallocate more resources to IPv4 routing, you can change the SDM template. Using the **show sdm prefer** command on SW2, as shown in Example 3-6, indicates that the current SDM template is "desktop default," which is the default template on a 3750E Catalyst switch. In this case, more resources need to be reserved for IPv4 routing; therefore, the template needs to be changed.

Example 3-6 ***show sdm prefer*** *Command Output on a Cisco Catalyst Switch*

```
SW2# show sdm prefer
 The current template is "desktop default" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 8 routed interfaces and 1024 VLANs.

  number of unicast mac addresses:                  6K
  number of IPv4 IGMP groups + multicast routes:    1K
  number of IPv4 unicast routes:                    8K
    number of directly-connected IPv4 hosts:        6K
    number of indirect IPv4 routes:                 2K
  number of IPv4 policy based routing aces:         0
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 0.875k
```

Using the global configuration command **sdm prefer**, as shown in Example 3-7, allows you to change the SDM template. In this case, the SDM template is being changed to **routing** so that more resources will be used for IPv4 unicast routing.

Example 3-7 *Changing the SDM Template on a Cisco 3750E Catalyst Switch*

```
SW2# config t
Enter configuration commands, one per line.  End with CNTL/Z.
DSW2(config)# sdm prefer ?
  access                          Access bias
  default                         Default bias
  dual-ipv4-and-ipv6              Support both IPv4 and IPv6
  indirect-ipv4-and-ipv6-routing  Supports more V4 and V6 Indirect Routes
  lanbase-routing                 Supports both IPv4 and IPv6 Static Routing
  routing                         Unicast bias
  vlan                            VLAN bias
DSW2(config)# sdm prefer routing
Changes to the running SDM preferences have been stored, but cannot take effect
until the next reload.
Use 'show sdm prefer' to see what SDM preference is currently active.
DSW2(config)# exit
DSW2# reload
System configuration has been modified. Save? [yes/no]: yes
Building configuration...
[OK]
Proceed with reload? [confirm]

%SYS-5-RELOAD: Reload requested by console. Reload Reason: Reload command.
```

After the reload, notice how the SDM template is listed as "desktop routing" in Example 3-8 and a larger amount of resources are dedicated to IPv4 indirect routes. However, also notice that while more resources are allocated to IPv4 unicast routes, fewer resources are allocated to other resources such as unicast MAC addresses.

Example 3-8 *Verifying That the SDM Template Was Changed After Reload*

```
SW2# show sdm prefer
 The current template is "desktop routing" template.
 The selected template optimizes the resources in
 the switch to support this level of features for
 8 routed interfaces and 1024 VLANs.

  number of unicast mac addresses:                  3K
  number of IPv4 IGMP groups + multicast routes:    1K
  number of IPv4 unicast routes:                    11K
    number of directly-connected IPv4 hosts:        3K
    number of indirect IPv4 routes:                 8K
  number of IPv4 policy based routing aces:         0.5K
  number of IPv4/MAC qos aces:                      0.5K
  number of IPv4/MAC security aces:                 1K
```

In Example 3-9, the output of **show platform tcam utilization** shows that the max masks/values are now 8144/8144 for IPv4 unicast indirectly connected routes when they were 2048 before. In addition, the used masks/values are now 3148, and therefore, the TCAM is able to forward traffic without having to punt the packets to the CPU.

Example 3-9 *Verifying the* ***tcam utilization*** *on the 3750E Catalyst Switch*

```
SW2# show platform tcam utilization

CAM Utilization for ASIC# 0                     Max            Used
                                              Masks/Values    Masks/values

 Unicast mac addresses:                        3292/3292         35/35
 IPv4 IGMP groups + multicast routes:          1120/1120          1/1
 IPv4 unicast directly-connected routes:       3072/3072          8/8
 IPv4 unicast indirectly-connected routes:     8144/8144         3148/3148
 IPv4 policy based routing aces:                490/490          13/13
 IPv4 qos aces:                                 474/474          21/21
 IPv4 security aces:                            964/964          42/42

Note: Allocation of TCAM entries per feature uses
a complex algorithm. The above information is meant
to provide an abstract view of the current TCAM utilization
```

High CPU Utilization Troubleshooting on a Switch

The load on a switch's CPU is often low, even under high utilization, thanks to the TCAM. Because the TCAM maintains a switch's forwarding logic at the data plane, the CPU is rarely tasked to forward traffic. The **show processes cpu** command can be used on a Cisco Catalyst switch to display CPU utilization levels, as demonstrated in Example 3-10.

Example 3-10 ***show processes cpu*** *Command Output on a Cisco Catalyst Switch*

```
SW1# show processes cpu
CPU utilization for five seconds: 19%/15%; one minute: 20%; five minutes: 13%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
   1           0         4          0  0.00%  0.00%  0.00%   0 Chunk Manager
   2           0       610          0  0.00%  0.00%  0.00%   0 Load Meter
   3         128         5      25600  0.00%  0.00%  0.00%   0 crypto sw pk pro
   4        2100       315       6666  0.00%  0.05%  0.05%   0 Check heaps
...OUTPUT OMITTED...
```

Notice in the output in Example 3-10 that the switch is reporting a 19 percent CPU load, with 15 percent of the CPU load used for interrupt processing. The difference between these two numbers is 4, suggesting that 4 percent of the CPU load is consumed with control plane processing.

Although such load utilization values might not be unusual for a router, these values might be of concern for a switch. Specifically, a typical CPU load percentage dedicated to interrupt processing is no more than 5 percent. A value as high as 10 percent is considered acceptable. However, the output given in Example 3-10 shows a 15 percent utilization, which is considered high for a Catalyst switch. Such a level implies that the switch's CPU is actively involved in forwarding packets that should normally be handled by the switch's TCAM. Of course, this value might be normal for your organization based on baseline information, even though according to Cisco it is a cause for concern. If the interrupt percent is greater than 10, you should take time to look into the reason why.

Periodic spikes in processor utilization are also not a major cause for concern if such spikes can be explained. Consider the following reasons that might cause a switch's CPU utilization to spike:

- The CPU is processing routing updates.
- The administrator is issuing a **debug** command (or other processor-intensive commands).
- Simple Network Management Protocol (SNMP) is being used to poll network devices.

If you determine that a switch's high CPU load is primarily the result of interrupts, you should examine the switch's packet-switching patterns and check the TCAM utilization.

If, however, the high CPU utilization is primarily the result of processes, you should investigate those specific processes.

A high CPU utilization on a switch might be a result of STP. Recall that an STP failure could lead to a broadcast storm, where Layer 2 broadcast frames endlessly circulate through a network. Therefore, when troubleshooting a performance issue, realize that a switch's high CPU utilization might be a symptom of another issue.

v Router Performance Issues

As we have seen, a Cisco Catalyst switch's performance can be the source of network problems. Similarly, a router performance issue can impact user data flowing through the network.

As an administrator, you might notice a sluggish response to Telnet sessions or SSH sessions that you attempt to establish with a router. Or, you might experience longer-than-normal ping response times from a router. Such symptoms might indicate a router performance issue. In these examples, the router's CPU is so busy it does not have time to respond to your Telnet session or the pings you have sent.

This section investigates three potential router issues, each of which might result in poor router performance. These three issues are

- Excessive CPU utilization
- The packet-switching mode of a router
- Excessive memory utilization

Excessive CPU Utilization

A router's processor (that is, CPU) utilization escalating to a high level but only remaining at that high level for a brief time could represent normal behavior. However, if a router's CPU utilization continually remains at a high level, network performance issues might result. Aside from latency that users and administrators can experience, a router whose CPU is overtaxed might not send routing protocol messages to neighboring routers in a timely fashion. As a result, routing protocol adjacencies can fail, resulting in some networks becoming unreachable.

Processes That Commonly Cause Excessive CPU Utilization

One reason that the CPU of a router might be overloaded is that the router is running a process that is taking up an unusually high percentage of its CPU resources. Following are four such processes that can result in excessive CPU utilization:

- **ARP Input process:** The ARP Input process is in charge of sending Address Resolution Protocol (ARP) requests. This process can consume an inordinate percentage of CPU resources if the router has to send numerous ARP requests. One

configuration that can cause such a high number of ARP requests is having a default route configured that points to an Ethernet interface. For example, perhaps a router had the **ip route 0.0.0.0 0.0.0.0 fastethernet 0/1** command entered in global configuration mode so that all packets with no explicit route in the routing table will be forwarded out Fa0/1. At first, this appears harmless; however, such a configuration should be avoided because an ARP Request has to be sent for every destination IP address in every packet that is received by the router and forwarded out Fa0/1. This is because the **ip route** command is stating that all IP addresses (0.0.0.0 0.0.0.0) are reachable through the directly connected interface fastethernet 0/1. Therefore, instead of ARPing for the MAC address of a next-hop IP address, you ARP for the MAC address of the IP address in each packet. That will result in an excessive number of ARP requests, which will cause strain on the CPU. In addition, many of the ARP requests will go unanswered and result in dropped packets. The better option is to specify the next-hop IP address because the router will only have to ARP for the MAC of the next-hop IP address when forwarding the packets out Fa0/1.

- **Net Background process:** An interface has a certain number of buffers available to store packets. These buffers are sometimes referred to as the queue of an interface. If an interface needs to store a packet in a buffer but all interface buffers are in use, the interface can pull from a main pool of buffers that the router maintains. The process that allows an interface to allocate one of these globally available buffers is Net Background. If the *throttles*, *ignored*, and *overrun* parameters are incrementing on an interface, the underlying cause might be the Net Background process consuming too many CPU resources.

- **IP Background process:** The IP Background process handles an interface changing its state. A state change might be an interface going from an Up state to a Down state, or vice versa. Another example of state change is an interface's IP address changing. Therefore, anything that can cause repeated state changes, such as bad cabling, might result in the IP Background process consuming a high percentage of CPU resources.

- **TCP Timer process:** The TCP Timer process runs for each TCP router connection. Therefore, many connections can result in high CPU utilization by the TCP Timer process, whether they are established or embryonic. An established TCP connection is one that has successfully completed the three-way handshake. An embryonic connection occurs when the TCP three-way handshake is only two-thirds completed. For example, the client sends the SYN packet to the server, and then the server sends a SYN/ACK back. At this point, the server is in the embryonic state (waiting for an ACK from the client to complete the three-way handshake and establish the connection). However, if the client does not send the ACK back, the server will sit in the embryonic state until it times out. This could be due to connectivity issues or malicious intent.

Cisco IOS Commands Used for Troubleshooting High Processor Utilization

Table 3-3 offers a collection of **show** commands that can be valuable when troubleshooting high CPU utilization on a router.

Table 3-3 *Commands for Troubleshooting High CPU Utilization*

Key Topic

Command	Description
show ip arp	Displays the ARP cache for a router. If several entries are in the Incomplete state, you might suspect a malicious scan (for example, a ping sweep) of a subnet, or you have a route pointing out an Ethernet interface as described in our ARP Input process discussion.
show interface [*interface-id*]	Displays a collection of interface statistics. If the throttles, overruns, or ignore counters continually increment, you might suspect that the Net Background process is attempting to allocate buffer space for an interface from the main buffer pool of the router.
show tcp statistics	Provides information about the number of TCP segments a router sends and receives, including the number of connections initiated, accepted, established, and closed. A high number of connections can explain why the TCP Timer process might be consuming excessive CPU resources. If you see an excessive number of embryonic connections, you might be under a DoS attack.
show processes cpu	Displays average CPU utilization over 5-second, 1-minute, and 5-minute intervals, in addition to listing all the router processes and the percentage of CPU resources consumed by each of those processes.
show processes cpu history	Displays a graphical view of CPU utilization over the past 60 seconds, 1 hour, and 3 days. This graphical view can indicate whether an observed high CPU utilization is a temporary spike in utilization or whether the high CPU utilization is an ongoing condition.

Example 3-11 shows sample output from the **show ip arp** command. In the output, only a single instance exists of an Incomplete ARP entry. However, a high number of such entries can suggest the scanning of network resources, which might indicate malicious reconnaissance traffic or that you have a route pointing out an Ethernet interface instead of to a next-hop IP address.

Example 3-11 ***show ip arp*** *Command Output*

```
R2# show ip arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.3.3.2                61  0009.b7fa.d1e0   ARPA   Ethernet0/0
Internet  10.3.3.1                 -  00d0.06fe.9ea0   ARPA   Ethernet0/0
Internet  192.168.1.50             0  Incomplete       ARPA
```

Example 3-12 shows sample output from the **show interface** [*interface-id*] command. Note the throttles, overrun, and ignored counters. If these counters continue to

increment, the Net Background process might be consuming excessive CPU resources while it allocates buffers from the main buffer pool of the router.

Example 3-12 ***show interface*** *interface-id Command Output*

```
R2# show interface e0/0
Ethernet0/0 is up, line protocol is up
  Hardware is AmdP2, address is 00d0.06fe.9ea0 (bia 00d0.06fe.9ea0)
  Internet address is 10.3.3.1/24
  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 00:00:02, output 00:00:02, output hang never
  Last clearing of "show interface" counters never
  Input queue: 0/75/0/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 0 bits/sec, 1 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     2156 packets input, 164787 bytes, 0 no buffer
     Received 861 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
     2155 packets output, 212080 bytes, 0 underruns
     0 output errors, 0 collisions, 7 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
```

Example 3-13 shows sample output from the **show tcp statistics** command. If the output indicates numerous connections, the TCP Timer process might be consuming excessive CPU resources while simultaneously maintaining all those connections. If you have a high number of initiated connections with a low number of established connections, it indicates that the three-way handshake is not being completed. This might be due to a DoS attack that is attempting to consume all the TCP connection slots.

Example 3-13 ***show tcp statistics*** *Command Output*

```
R2# show tcp statistics
Rcvd: 689 Total, 0 no port
      0 checksum error, 0 bad offset, 0 too short
      474 packets (681 bytes) in sequence
      0 dup packets (0 bytes)
      0 partially dup packets (0 bytes)
      0 out-of-order packets (0 bytes)
```

```
      0 packets (0 bytes) with data after window
      0 packets after close
      0 window probe packets, 0 window update packets
      1 dup ack packets, 0 ack packets with unsend data
      479 ack packets (14205 bytes)
Sent: 570 Total, 0 urgent packets
      1 control packets (including 0 retransmitted)
      562 data packets (14206 bytes)
      0 data packets (0 bytes) retransmitted
      0 data packets (0 bytes) fastretransmitted
      7 ack only packets (7 delayed)
      0 window probe packets, 0 window update packets
0 Connections initiated, 1 connections accepted, 1 connections established
0 Connections closed (including 0 dropped, 0 embryonic dropped)
0 Total rxmt timeout, 0 connections dropped in rxmt timeout
0 Keepalive timeout, 0 keepalive probe, 0 Connections dropped in keepalive
```

Example 3-14 shows sample output from the **show processes cpu** command. The output in this example indicates a 34 percent CPU utilization in the past 5 seconds, with 13 percent of CPU resources being spent on interrupts. The output also shows the 1-minute CPU utilization average as 36 percent and the 5-minute average as 32 percent. Individual processes running on the router are also shown, along with their CPU utilization levels. Note the ARP Input, Net Background, TCP Timer, and IP Background processes referred to in this section.

Example 3-14 ***show processes cpu*** *Command Output*

```
R2# show processes cpu
CPU utilization for five seconds: 34%/13%; one minute: 36%; five minutes: 32%
PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min   TTY Process
...OUTPUT OMITTED...
  12           4       69         57  0.00%  0.00%  0.00%    0 ARP Input
  13           0        1          0  0.00%  0.00%  0.00%    0 HC Counter Timer
  14           0        5          0  0.00%  0.00%  0.00%    0 DDR Timers
  15          12        2       6000  0.00%  0.00%  0.00%    0 Entity MIB API
  16           4        2       2000  0.00%  0.00%  0.00%    0 ATM Idle Timer
  17           0        1          0  0.00%  0.00%  0.00%    0 SERIAL A'detect
  18           0     3892          0  0.00%  0.00%  0.00%    0 GraphIt
  19           0        2          0  0.00%  0.00%  0.00%    0 Dialer event
  20           0        1          0  0.00%  0.00%  0.00%    0 Critical Bkgnd
  21         132      418        315  0.00%  0.00%  0.00%    0 Net Background
  22           0       15          0  0.00%  0.00%  0.00%    0 Logger
...OUTPUT OMITTED...
  46           0      521          0  0.00%  0.00%  0.00%    0 SSS Test Client
  47          84      711        118  0.00%  0.00%  0.00%    0 TCP Timer
  48           4        3       1333  0.00%  0.00%  0.00%    0 TCP Protocols
```

```
 49           0      1           0  0.00%  0.00%  0.00%   0 Socket Timers
 50           0      15          0  0.00%  0.00%  0.00%   0 HTTP CORE
 51          12      5        2400  0.00%  0.00%  0.00%   0 PPP IP Route
 52           4      5         800  0.00%  0.00%  0.00%   0 PPP IPCP
 53         273     157        1738  0.00%  0.00%  0.00%   0 IP Background
 54           0      74          0  0.00%  0.00%  0.00%   0 IP RIB Update
...OUTPUT OMITTED...
```

Example 3-15 shows sample output from the **show processes cpu** history command. The graphical output produced by this command is useful in determining whether a CPU spike is temporary or whether it is an ongoing condition.

Example 3-15 ***show processes cpu history*** *Command Output*

```
R2# show processes cpu history


     4                    11111     4444411111          11111
     9444445555544444444447777755555888888888877777555555777775555
  100
   90
   80
   70
   60
   50 *                              *****
   40 *                              *****
   30 *                              *****
   20 *                    *****     **********          *****
   10 *     *****          ************************************
     0....5....1....1....2....2....3....3....4....4....5....5....6
               0    5    0    5    0    5    0    5    0    5    0
               CPU% per second (last 60 seconds)




     6111111111121112211311111111111111121111111111111211111111111
     3765778462816371177566657715737672176743747376640089277752 77
  100
   90
   80
   70
   60 *
   50 *
   40 *                   *
   30 *                   *
   20 ******* * *** ************ ** *** ****  * * *** * ** ****
```

```
 10 ############################################################
   0....5....1....1....2....2....3....3....4....4....5....5....6
             0    5    0    5    0    5    0    5    0    5    0
             CPU% per minute (last 60 minutes)
            * = maximum CPU%   # = average CPU%




    56434334644444334443442544453443
    46868692519180723579483247519306
100
 90
 80
 70  *      *
 60  *      *
 50 *** *   * * *    **  * * ***
 40 ********** ****** *********
 30 ********** *********
 20 ********************************
 10 ################################
   0....5....1....1....2....2....3....3....4....4....5....5....6....6....7..
             0    5    0    5    0    5    0    5    0    5    0    5    0
                  CPU% per hour (last 72 hours)
                 * = maximum CPU%   # = average CPU%
```

Understanding Packet-Switching Modes (Routers and Multilayer Switches)

In addition to the high CPU utilization issues previously discussed, a router's packet-switching mode can impact router performance. Before discussing the most common switching modes, realize that the way a router handles packets (or is capable of handling packets) largely depends on the router's architecture. Therefore, for real-world troubleshooting, consult the documentation for your router to determine how it implements packet switching.

In general, however, Cisco routers and multilayer switches support the following three primary modes of packet switching:

- Process switching
- Fast switching (Route Caching)
- Cisco Express Forwarding (Topology-Based Switching)

Packet switching involves the router making a decision about how a packet should be forwarded and then forwarding that packet out of the appropriate router interface.

Operation of Process Switching

When a router routes a packet (that is, performs packet switching), the router removes the packet's Layer 2 header, examines the Layer 3 addressing, and decides how to forward the packet. The Layer 2 header is then rewritten (which involves changing the source and destination MAC addresses and computing a new cyclic redundancy check [CRC]), and the packet is forwarded out of the appropriate interface. With process switching, as illustrated in Figure 3-4, the router's CPU becomes directly involved with packet-switching decisions. As a result, the performance of a router configured for process switching can suffer significantly.

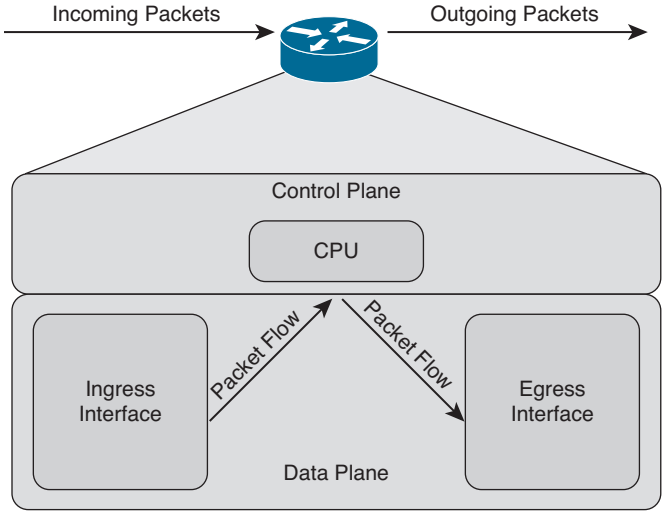


Figure 3-4 *Data Flow with Process Switching*

An interface can be configured for process switching by disabling fast switching and CEF on that interface. The interface configuration mode command used to disable fast switching and CEF at the same time is **no ip route-cache**.

Operation of Fast Switching (Route Caching)

Fast switching uses a fast cache maintained in a router's data plane. The fast cache contains information about how traffic from different data flows should be forwarded. As seen in Figure 3-5, the first packet in a data flow is process-switched by a router's CPU. After the router determines how to forward the first frame of a data flow, that forwarding information is stored in the fast cache. Subsequent packets in that same data flow are forwarded based on information in the fast cache, as opposed to being process-switched. As a result, fast switching reduces a router's CPU utilization more than process switching does.

Fast switching can be enabled by turning off CEF in interface configuration mode with the **no ip route-cache cef** command.

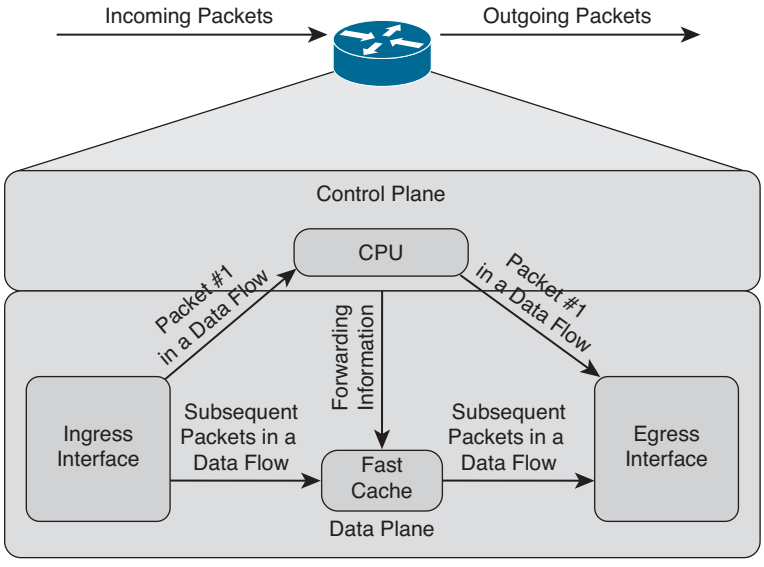


Figure 3-5 *Data Flow with Fast Switching*

Operation of Cisco Express Forwarding (Topology-Based Switching)

Cisco Express Forwarding (CEF) maintains two tables in the data plane. Specifically, the Forwarding Information Base (FIB) maintains Layer 3 forwarding information, whereas the Adjacency Table maintains Layer 2 information for next hops listed in the FIB.

Using these tables, populated from a router's IP routing table and ARP cache, CEF can efficiently make forwarding decisions. Unlike fast switching, CEF does not require the first packet of a data flow to be process-switched. Rather, an entire data flow can be forwarded at the data plane, as seen in Figure 3-6.

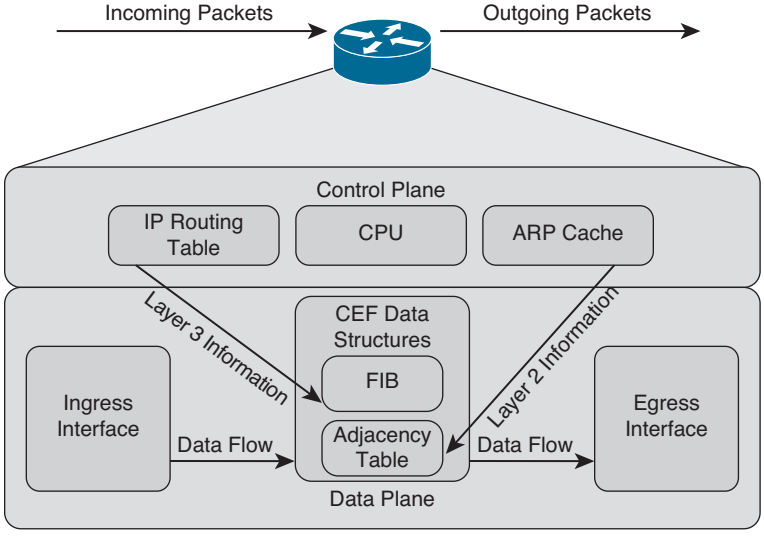


Figure 3-6 *Data Flow with Cisco Express Forwarding*

On many router platforms, CEF is enabled by default. If it is not, you can globally enable it with the **ip cef** command. Alternatively, you can enable CEF for a specific interface with the interface configuration mode command **ip route-cache cef**.

Date Night Example of Process-Switching Modes

Let's pretend that my wife and I are going out to dinner and we are leaving our two children with a babysitter. If we are "Process Switching" with the babysitter, every time our children ask the babysitter for a cookie, she has to call us to ask for permission to give the children a cookie. If the children ask ten times, she has to call us ten times. If we are "Fast Switching" with the babysitter, the first time she calls us, we say yes and then create a "route cache" for the babysitter that states, "if the kids want more, just give them more without calling us." Finally, if we are using "CEF" with the babysitter, before we leave for dinner, we take out the cookie jar, place it on the counter, and tell her to have an awesome evening with the kids. As you can see from this example, date night is better when we use CEF.

Troubleshooting Packet-Switching Modes

Table 3-4 provides a selection of commands that you can use when troubleshooting the packet-switching modes of a router.

Table 3-4 *Commands for Troubleshooting a Router's Packet-Switching Modes*

Command	Description
show ip interface [*interface_id*]	Displays multiple interface statistics, including information about the packet-switching mode of an interface.
show ip cache	Displays the contents of the route-cache from a router if fast switching is enabled.
show processes cpu | include IP Input	Displays information about the IP input process on a router. The CPU utilization for this process might show a high value if the CPU of a router is actively engaged in process-switching traffic because you turned off fast switching and CEF.
show ip cef	Displays the contents of a router's FIB.
show ip cef adjacency *egress-interface-id next-hop-ip-address* **detail**	Displays destinations reachable through the combination of the specified egress interface and next-hop IP address.
show adjacency detail	Provides information contained in the adjacency table of a router, including protocol and timer information.

Example 3-16 shows sample output from the **show ip interface** *interface-id* command. The output indicates that fast switching and CEF switching are enabled on interface FastEthernet 0/0. The reference to Flow switching being disabled refers to the Cisco IOS NetFlow feature, which you can use to collect traffic statistics.

Example 3-16 ***show ip interface*** *interface-id Command Output*

```
R4# show ip interface fa 0/0
FastEthernet0/0 is up, line protocol is up
...OUTPUT OMITTED...
  ICMP mask replies are never sent
  IP fast switching is enabled
  IP fast switching on the same interface is disabled
  IP Flow switching is disabled
  IP CEF switching is enabled
  IP CEF Fast switching turbo vector
  IP multicast fast switching is enabled
  IP multicast distributed fast switching is disabled
  IP route-cache flags are Fast, CEF
...OUTPUT OMITTED...
```

Example 3-17 shows sample output from the **show ip cache** command. If fast switching is enabled and CEF is disabled, a router begins to populate its route cache. This command shows the contents of a router's route cache.

Example 3-17 ***show ip cache*** *Command Output*

```
R4# show ip cache
IP routing cache 3 entries, 588 bytes
   12 adds, 9 invalidates, 0 refcounts
Minimum invalidation interval 2 seconds, maximum interval 5 seconds,
   quiet interval 3 seconds, threshold 0 requests
Invalidation rate 0 in last second, 0 in last 3 seconds
Last full cache invalidation occurred 04:13:57 ago

Prefix/Length           Age       Interface        Next Hop
10.8.8.4/32             00:00:07   FastEthernet0/1  10.8.8.4
10.8.8.6/32             00:00:10   FastEthernet0/1  10.8.8.6
192.168.0.0/24          00:00:10   FastEthernet0/0  10.3.3.1
```

Example 3-18 shows sample output from the **show processes cpu | include IP Input** command. In the output, the IP input process was using only 0.08 percent of its router's CPU capacity during the last 5-second interval. However, a high percentage value might indicate that a router was performing process switching, where the CPU was directly involved in packet switching.

Example 3-18 ***show processes cpu | include IP Input*** *Command Output*

```
R4# show processes cpu | include IP Input
  63        3178      7320        434  0.08%  0.06%  0.04%   0 IP Input
```

Example 3-19 shows sample output from the **show ip cef** command. The output contains the contents of the FIB for a router. Notice that the prefix is listed, followed by the next hop that will be used to reach the prefix, and then the interface that will be used to reach it. Note that if a next hop of the network prefix is set to *receive,* that network/IP is local to the router, and any packets destined to that specific IP will be processed by the CPU of the router. Examining the output closely, you will see that the receive entries are subnet IDs, local host IP addresses, and broadcast addresses, ensuring that they are processed by the router and not forwarded. The *attached* next hop indicates that the network is a directly connected route on the router.

Example 3-19 ***show ip cef*** *Command Output*

```
R4# show ip cef
Prefix              Next Hop            Interface
0.0.0.0/0           drop                Null0 (default route handler entry)
0.0.0.0/32          receive
10.1.1.0/24         10.3.3.1            FastEthernet0/0
10.1.1.2/32         10.3.3.1            FastEthernet0/0
10.3.3.0/24         attached            FastEthernet0/0
10.3.3.0/32         receive
10.3.3.1/32         10.3.3.1            FastEthernet0/0
10.3.3.2/32         receive
10.3.3.255/32       receive
10.4.4.0/24         10.3.3.1            FastEthernet0/0
10.5.5.0/24         10.3.3.1            FastEthernet0/0
10.7.7.0/24         10.3.3.1            FastEthernet0/0
10.7.7.2/32         10.3.3.1            FastEthernet0/0
10.8.8.0/24         attached            FastEthernet0/1
10.8.8.0/32         receive
10.8.8.1/32         receive
10.8.8.4/32         10.8.8.4            FastEthernet0/1
10.8.8.5/32         10.8.8.5            FastEthernet0/1
10.8.8.6/32         10.8.8.6            FastEthernet0/1
10.8.8.7/32         10.8.8.7            FastEthernet0/1
10.8.8.255/32       receive
192.168.0.0/24      10.3.3.1            FastEthernet0/0
224.0.0.0/4         drop
224.0.0.0/24        receive
255.255.255.255/32   receive
```

Example 3-20 shows sample output from the **show ip cef adjacency** *egress-interface-id next-hop-IP-address* **detail** command. This command shows the IP addresses that the router knows how to reach using the specified combination of next-hop IP address and egress interface. In this example, 10.8.8.6 is the IP address of a host and not a router. Therefore, no other IP addresses are known to have a next-hop IP address of 10.8.8.6 with an egress interface of Fast Ethernet 0/1.

Example 3-20 ***show ip cef adjacency*** *egress-interface-id next-hop-IP-address* ***detail*** *Command Output*

```
R4# show ip cef adjacency fa 0/1 10.8.8.6 detail
IP CEF with switching (Table Version 25), flags=0x0
  25 routes, 0 reresolve, 0 unresolved (0 old, 0 new), peak 0
  25 leaves, 21 nodes, 25640 bytes, 90 inserts, 65 invalidations
  0 load sharing elements, 0 bytes, 0 references
  universal per-destination load sharing algorithm, id 24360DB1
  5(2) CEF resets, 1 revisions of existing leaves
  Resolution Timer: Exponential (currently 1s, peak 1s)
  0 in-place/0 aborted modifications
  refcounts:  5702 leaf, 5632 node

  Table epoch: 0 (25 entries at this epoch)

Adjacency Table has 5 adjacencies
10.8.8.6/32, version 10, epoch 0, cached adjacency 10.8.8.6
0 packets, 0 bytes
  via 10.8.8.6, FastEthernet0/1, 0 dependencies
    next hop 10.8.8.6, FastEthernet0/1
    valid cached adjacency
```

Example 3-21 shows sample output from the **show adjacency detail** command. When you see a particular adjacency listed in the FIB, you can issue this command to confirm that the router has information about how to reach that adjacency. In this case, if we need to send a packet to 10.3.3.1, we will send the packet out FastEthernet0/0, which requires a Layer 2 frame with a source and destination MAC address. These MAC addresses are already listed in the adjacency table. The value 00D006FE9EA00009B7FAD1E00800 can be broken into three parts:

- 00D006FE9EA0 = Destination MAC address
- 0009B7FAD1E0 = Source MAC address
- 0800 = Well-know Ethertype value for IP

Example 3-21 ***show adjacency detail*** *Command Output*

```
R4# show adjacency detail
Protocol Interface                 Address
IP       FastEthernet0/0           10.3.3.1(19)
                                   32 packets, 1920 bytes
                                   00D006FE9EA00009B7FAD1E00800
                                   ARP        03:53:01
                                   Epoch: 0
IP       FastEthernet0/1           10.8.8.6(5)
                                   4 packets, 264 bytes
                                   0008A3B895C40009B7FAD1E10800
                                   ARP        03:53:35
                                   Epoch: 0
...OUTPUT OMITTED...
```

Now that you have reviewed the different packet-switching options for a router, you can better analyze how a router is forwarding specific traffic. Following is a list of troubleshooting steps that you can follow if you suspect that network traffic is being impacted by a performance problem on one of the routers along the path from the source to the destination.

Step 1. Use the **traceroute** command to determine which router along the path is causing excessive delay.

Step 2. After you identify a router that is causing unusually high delay, use the **show processes cpu** command to see the CPU utilization of that router and identify any processes that might be consuming an unusually high percentage of the CPU.

Step 3. Use the **show ip route** *ip-address* command to verify that the router has a route to the destination IP address.

Step 4. Use the **show ip cef** command to determine whether all the router interfaces are configured to use CEF.

Step 5. Use the **show ip cef** *ip-address* **255.255.255.255** command to verify that CEF has an entry in its FIB that can reach the specified IP address. Part of the output from this command will be the next-hop adjacency to which traffic should be forwarded, along with the egress interface used to send traffic to that next hop.

Step 6. Issue the **show adjacency** *interface-id* **detail** command to verify that CEF has an entry in its adjacency table for the egress interface identified in Step 5.

Step 7. With the **show ip arp** command, you can then confirm that the router knows the MAC address associated with the next-hop IP address shown in the output from Step 6.

Step 8. You can then connect to the next-hop device and verify that the MAC address identified in Step 7 is indeed correct.

You can repeat these steps on the next-hop device or on another router whose response time displayed in the output from Step 1 is suspect.

Excessive Memory Utilization

Much like a PC, router performance can suffer if it lacks sufficient available memory. For example, perhaps you install a version of Cisco IOS on a router, and that router does not have the minimum amount of memory required to support that specific Cisco IOS image. Even though the router might load the image and function, its performance might be sluggish. Assuming that a router does have the recommended amount of memory for its installed Cisco IOS image, consider the following as potential memory utilization issues.

Memory Leak

When a router starts a process, that process can allocate a block of memory. When the process completes, the process should return its allocated memory to the router's pool of memory. If not all the allocated memory is returned to the router's main memory pool, a memory leak occurs. Such a condition usually results from a bug in the Cisco IOS version running on the router, requiring an upgrade of the router's Cisco IOS image.

Example 3-22 shows sample output from the **show memory allocating-process totals** command. This command can help identify memory leaks. The output shows information about memory availability on a router after the Cisco IOS image of the router has been decompressed and loaded and the total amount of memory that is being used by the various processes.

Example 3-22 ***show memory allocating-process totals*** *Command Output*

```
R4# show memory allocating-process totals
                Head    Total(b)     Used(b)     Free(b)   Lowest(b)  Largest(b)
Processor   83D27480    67463064    15347168    52115896    50311080    50127020
      I/O    7C21800     4057088     2383016     1674072     1674072     1674044

Allocator PC Summary for: Processor

    PC          Total   Count  Name
0x809D7A30    1749360     180  Process Stack
0x80A7F664     918024      10  Init
0x81CEF6A0     882576       4  pak subblock chunk
0x81C04D9C     595344      54  TCL Chunks
0x800902A4     490328       6  MallocLite
...OUTPUT OMITTED...
```

The *Head* column in the output refers to the address (in hexadecimal) of the memory allocation chain. The *Total* column is the total amount of memory available in bytes. The *Used* column indicates how much has been used, and *Free* indicates how much

is remaining. The *Lowest* column shows the lowest amount of free memory (in bytes) that has been available since the router last booted. The *Largest* column indicates the largest block of available memory. Following this summary information, the output shows detailed memory allocation information for each process running on a router. If a process is consuming a larger-than-normal amount of memory, it is likely because of a memory leak. A memory leak occurs when a process does not free the memory that it is finished using. Therefore, the block of memory remains reserved and will only be released when the router is reloaded. Typically, memory leaks are a result of bugs or poor coding in the Cisco IOS Software. The best solution is to upgrade the Cisco IOS Software to a version that fixes the issue.

Memory Allocation Failure

A memory allocation failure (which produces a MALLOCFAIL error message) occurs when a process attempts to allocate a block of memory and fails to do so. One common cause for a MALLOCFAIL error is a security issue. For example, a virus or a worm that has infested the network can result in a MALLOCFAIL error. Alternatively, a MALLOCFAIL error might be the result of a bug in the router's version of Cisco IOS. You can use the Cisco Bug Toolkit (available from www.cisco.com/cgi-bin/Support/Bugtool/launch_bugtool.pl) to research any such known issues with the version of Cisco IOS running on a router. Personally, I have witnessed the MALLOCFAIL error message when using an Integrated Services Router (ISR) that was running NAT, and another instance when I tried to load the complete IPS Signature Definition File on another ISR when I knew it could not handle it.

Buffer Leak

Similar to a memory leak, in which a process does not return all of its allocated memory to the router upon terminating, a buffer leak occurs when a process does not return a buffer to the router when the process has finished using the buffer. Consider the output of the **show interfaces** command seen in Example 3-23.

Example 3-23 *Identifying a Wedged Interface*

```
R4# show interfaces
...OUTPUT OMITTED...
  Input queue: 76/75/780/0 (size/max/drops/flushes); Total output drops: 0
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
...OUTPUT OMITTED...
```

Notice the numbers 76 and 75 highlighted in the output. These values indicate that an input queue of the interface has a capacity of 75 packets, and that queue currently has 76 packets. This is an oversubscription of the queue space. An interface in this condition is called a *wedged interface*. In such a condition, the router does not forward traffic coming into the wedged interface.

The **show buffers** command can also be helpful in diagnosing a buffer leak. To illustrate, consider the output of the **show buffers** command shown in Example 3-24.

Example 3-24 ***show buffers*** *Command Output*

```
R4# show buffers
Buffer elements:
     1118 in free list (500 max allowed)
     570 hits, 0 misses, 1119 created

Public buffer pools:
Small buffers, 104 bytes (total 71, permanent 50, peak 71 @ 00:21:43):
     53 in free list (20 min, 150 max allowed)
     317 hits, 7 misses, 0 trims, 21 created
     0 failures (0 no memory)
Middle buffers, 600 bytes (total 49, permanent 25, peak 49 @ 00:21:43):
     5 in free list (10 min, 150 max allowed)
     122 hits, 8 misses, 0 trims, 24 created
...OUTPUT OMITTED...
```

This output indicates that the router has 49 middle buffers, but only 5 of those 49 buffers are available. Such a result might indicate a process allocating buffers but failing to deallocate them. Like a memory leak, a buffer leak might require updating the Cisco IOS image of a router.

Excessive BGP Memory Use

If a router is running Border Gateway Protocol (BGP), be aware that BGP runs multiple processes and can consume significant amounts of router memory. The **show processes memory | include BGP** command, as shown in Example 3-25, can show you how much memory is being consumed by the various BGP processes of a router. If BGP is consuming a large percentage of your router memory, you might consider filtering out unneeded BGP routes, upgrading the memory on that router, or running BGP on a different platform that has more memory.

Example 3-25 ***show processes memory | include BGP*** *Command Output*

```
R1# show processes memory | include BGP|^ PID
PID TTY  Allocated      Freed    Holding    Getbufs    Retbufs Process
 184   0          0          0       7096          0          0 BGP Task
 198   0          0          0      10096          0          0 BGP Scheduler
 229   0      38808          0      11520          0          0 BGP Router
 231   0          0          0      10096          0          0 BGP I/O
 262   0          0          0      10096          0          0 BGP Scanner
 284   0          0          0       7096          0          0 BGP Event
```

Depending on the router platform, your router might have multiple line cards with different amounts of memory available on each line card. The **show diag** command can help you isolate a specific line card that is running low on memory, perhaps because that line card is running BGP.

Exam Preparation Tasks

As mentioned in the section "How to Use This Book" in the Introduction, you have several choices for exam preparation: the exercises here; Chapter 22, "Final Preparation"; and the exam simulation questions on the CD-ROM.

Review All Key Topics

Review the most important topics in this chapter, noted with the Key Topics icon in the outer margin of the page. Table 3-5 lists a reference of these key topics and the page numbers on which each is found.

Table 3-5 *Key Topics for Chapter 3*

Key Topic Element	Description	Page Number
List	Components in a Catalyst switch	6
Table 3-2	Errors in the **show interfaces** *interface_id* **counters errors** Command	8
List	Reasons why a packet could be punted from a switch's TCAM to its CPU	12
Section	High CPU Utilization Troubleshooting on a Switch	16
List	Identifies processes that cause excessive router CPU utilization	17
Table 3-3	Commands for Troubleshooting High CPU Utilization	19
List	Three primary modes of packet switching	23
Table 3-4	Commands for Troubleshooting a Router's Packet-Switching Modes	26
Step list	Example of troubleshooting the forwarding of packets	30
Section	Excessive Memory Utilization	31

Define Key Terms

Define the following key terms from this chapter and check your answers in the glossary:

backplane, control plane, forwarding logic, ingress port, egress port, half-duplex, full-duplex, TCAM, ARP Input process, Net Background process, IP Background process, TCP Timer process, Process switching, Fast switching, CEF, memory leak, memory allocation failure, buffer leak

Command Reference to Check Your Memory

This section includes the most important EXEC commands covered in this chapter. It might not be necessary to memorize the complete syntax of every command, but you should be able to remember the basic keywords that are needed.

To test your memory of the commands, cover the right side of Table 3-6 with a piece of paper, read the description on the left side, and then see how much of the command you can remember.

The 300-135 TSHOOT exam focuses on practical, hands-on skills that are used by a networking professional. Therefore, you should be able to identify the commands needed to verify router and switch configurations.

Table 3-6 ***EXEC*** *Commands*

Task	Command Syntax
A Cisco Catalyst 3750E Series switch command that can be used to verify the maximum and used TCAM resources for various services and features on the switch.	**show platform tcam utilization**
A Cisco Catalyst switch command that can be used to display the current SDM template being used on the switch.	**show sdm prefer**
Displays a router's ARP cache. (Note: If a large number of the entries are in the Incomplete state, you might suspect a malicious scan [for example, a ping sweep] of a subnet.)	**show ip arp**
Shows a collection of interface statistics. (Note: If the throttles, overruns, or ignore counters continually increment, you might suspect that the Net Background process is attempting to allocate buffer space for an interface from the router's main buffer pool.)	**show interface** [*interface-id*]
Provides information about the number of TCP segments a router sends and receives, including the number of connections initiated, accepted, established, and closed. (Note: A high number of connections might explain why the TCP Timer process is consuming excessive CPU resources.)	**show tcp statistics**

Task	Command Syntax
Displays average CPU utilization over 5-second, 1-minute, and 5-minute intervals, in addition to listing all the router processes and the percentage of CPU resources consumed by each of those processes.	**show processes cpu**
Shows a graphical view of CPU utilization over the past 60 seconds, 1 hour, and 3 days. (Note: This graphical view can indicate whether an observed high CPU utilization is a temporary spike in utilization or whether the high CPU utilization is an ongoing condition.)	**show processes cpu history**
Displays multiple interface statistics, including information about the packet-switching mode of an interface.	**show ip interface** [*interface-id*]
Shows the contents of the fast cache for a router if fast switching is enabled.	**show ip cache**
Displays information about the IP Input process on a router. (Note: The CPU utilization for this process might show a high value if the CPU of a router is actively engaged in process-switching traffic.)	**show processes cpu | include IP Input**
Displays the router's Layer 3 forwarding information, in addition to multicast, broadcast, and local IP addresses.	**show ip cef**
Verifies that a valid adjacency exists for a connected host.	**show adjacency**
Displays destinations reachable through the combination of the specified egress interface and next-hop IP address.	**show ip cef adjacency** *egress-interface-id next-hop-ip-address* **detail**
Provides information contained in a router's adjacency table, including protocol and timer information.	**show adjacency detail**
Displays information about packets forwarded by the router using a packet-switching mechanism other than CEF.	**show cef not-cef-switched**
Shows information about memory availability on a router after the router's Cisco IOS image has been decompressed and loaded. (Note: This command can help identify memory leaks.)	**show memory allocating-process totals**
Shows how many buffers (of various types) are currently free. (Note: This command can be helpful in diagnosing a buffer leak.)	**show buffers**
Shows how much memory is being consumed by the various BGP processes of a router.	**show processes memory | include bgp**
Shows the memory available on the line cards of a router.	**show diag**