Cisco Press

CCNP Practical Studies: Layer 3 Switching

Date: Nov 26, 2003 Sample Chapter is provided courtesy of Cisco Press.
Learn about Layer 3 switching in detail and how to configure Layer 3 switching on the flagship of the Cisco Catalyst product family, the Catalyst 6000/6500.

In the previous chapter, you were introduced to the concept of Layer 3 switching on the Catalyst 3550; however, the focus of the chapter was more on the fundamental concept of inter-VLAN routing and routing protocols rather than focusing specifically on Layer 3 switching. In this chapter, you learn about Layer 3 switching in detail and how to configure Layer 3 switching on the flagship of the Cisco Catalyst product family, the Catalyst 6000/6500.

NOTE

Although this chapter shows you how to configure Layer 3 switching on the Catalyst 6000/6500, the same concepts and configurations discussed in scenarios based around Layer 3 switching using Cisco Express Forwarding (CEF) can be applied to other CEF-based Cisco Catalyst Layer 3 switching platforms, such as the Catalyst 3550 and Catalyst 4000/4500 Supervisor 3/4 engines.

This chapter looks initially at software-only versus hardware-assisted Layer 3 (L3) switching (routing), examining the architectures used by each, which enables you to understand the limitations of software-based L3 switching and the advantages of hardware-based L3 switching. You learn about Multilayer switching (MLS), which represents an older Layer 3 switching technology used on older Catalyst switches and then learn about CEF-based Layer 3 switching, which is the current Layer 3 switching technology used on all next-generation Cisco Layer 3 switches (e.g., Catalyst 3550, Catalyst 4000/4500 Supervisor 3/4, and Catalyst 6000/6500 Supervisor 2 with PFC-2 + MSFC-2). You also learn about the architecture of the Catalyst 6000/6500, which represents the flagship of the Cisco Catalyst switching family.

Finally, the scenarios for this chapter are presented, which focus initially on MLS and then focus on the Catalyst 6000/6500 and how to configure CEF-based L3 switching on these switches. You also learn how to convert a Catalyst 6000/6500 from hybrid mode (CatOS) to native Cisco IOS, which is the future operating system for all Catalyst switches. After introductory material, the following scenarios are presented in this chapter:

- Scenario 6-1: Configuring MLS on the Catalyst 6000
- Scenario 6-2: Configuring CEF-based Layer 3 switching on the Catalyst 6000/6500 operating in hybrid mode
- Scenario 6-3: Upgrading from hybrid mode to native mode on the Catalyst 6000/6500
- Scenario 6-4: Configuring CEF-based Layer 3 switching on the Catalyst 6000/6500 operating in native mode

Introduction to Layer 3 Switching

In the previous chapter, you were introduced to the concept of inter-VLAN routing, which is required to enable hosts that belong to different VLANs on the same LAN network to communicate with each other. Implementing inter-VLAN routing introduces several benefits, which include the following:

- Reduces broadcast domains, increasing network performance and efficiency.

- Multilayer topologies based upon inter-VLAN routing are much more scalable and implement more efficient mechanisms for accommodating redundant paths in the network than equivalent flat Layer 2 topologies that rely on spanning tree alone.
- Allows for centralized security access control between each VLAN.
- Increases manageability by creating smaller "troubleshooting domains," where the effect of a faulty network interface card (NIC) is isolated to a specific VLAN rather than the entire network.

Of course, all of these features must be provided with a very important caveat—inter-VLAN routing should not affect performance, as users expect high performance from the LAN.

A popular approach to providing the benefits of inter-VLAN routing and also ensuring the performance of the LAN is not degraded has been to implement Layer 3 switches, which are essentially Layer 2 switches with a routing engine that is designed to specifically route traffic between VLANs in a LAN environment. Using Layer 3 switches for inter-VLAN routing as opposed to traditional routers is popular (and recommended) for the following reasons:

- **Performance versus Cost**—Layer 3 switches are much more cost effective than routers for delivering high-speed inter-VLAN routing. High performance routers are typically much more expensive than Layer 3 switches. For example, a Catalyst 3550-24 EMI switch sets you back $4,990 U.S. list, which provides a packet forwarding rate of 6.6 million packets per second with 24 * 10/100BASE-T ports and 2 * 1000BASE-X ports. A Cisco 7300 router with an NSE-100 engine provides a packet forwarding rate of 3.5 million packets per second, but sets you back $22,000 U.S. list and has only 2 * 1000BASE-T ports in its base configuration. Of course, the Cisco 7300 router has many more features and can support a wide variety of WAN media options; however, many of these extra features are not required for inter-VLAN routing.
- **Port density**—Layer 3 switches are enhanced Layer 2 switches and, hence, have the same high port densities that Layer 2 switches have. Routers on the other hand typically have a much lower port density.
- **Flexibility**—Layer 3 switches allow you to mix and match Layer 2 and Layer 3 switching, meaning you can configure a Layer 3 switch to operate as a normal Layer 2 switch, or enable Layer 3 switching as required.

Layer 3 switching is cheap because Layer 3 switches are targeted specifically for inter-VLAN routing, where only Ethernet access technologies are used in high densities. This makes it easy for Layer 3 switch vendors such as Cisco to develop high performance Layer 3 switches, as vendors can develop hardware chips (known as *application-specific integrated circuits* or *ASICs*) that specifically route traffic between Ethernet networks, without having to worry about the complexities of also supporting WAN technologies such as Frame Relay or ATM. Routing over WAN networks can still be supported, simply by plugging a traditional router that connects to the WAN networks into the LAN network. Figure 6-1 illustrates the concept of Layer 3 switching.

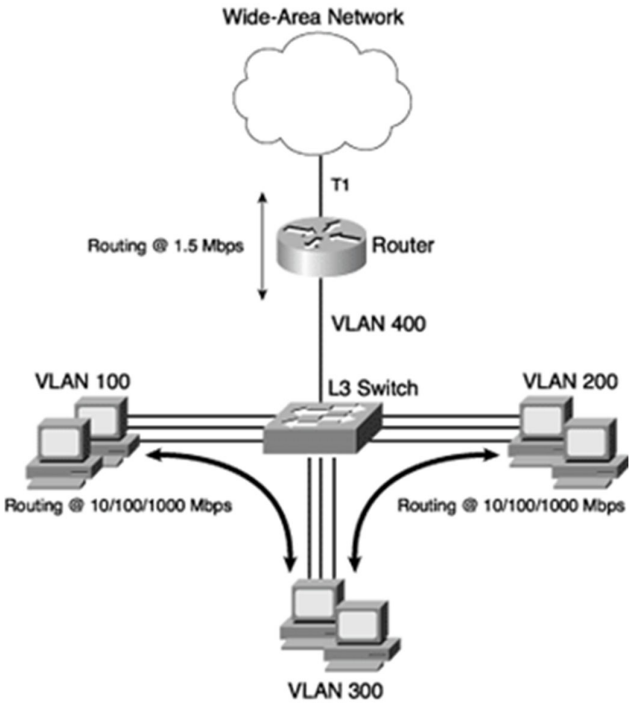


Figure 6-1 Layer 3 Switching

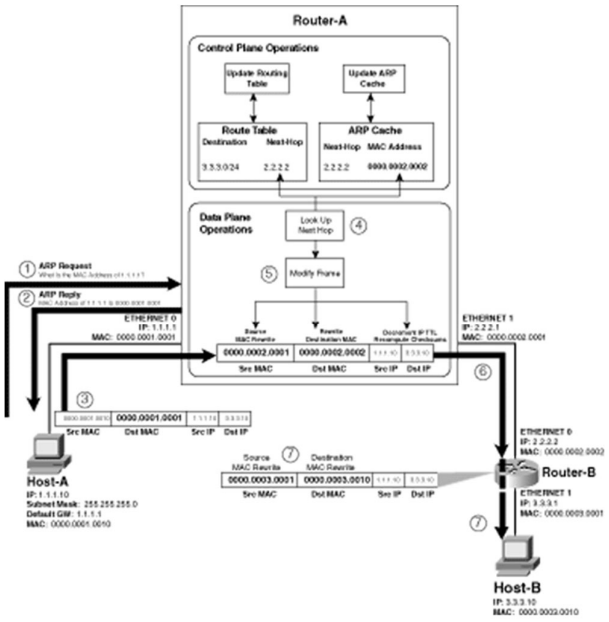
In Figure 6-1, a L3 switch provides switched LAN connections for each device in the network. Three user VLANs are present, and a routing engine on the L3 switch enables communications between each VLAN. The L3 switch possesses specialized hardware chips called application-specific integrated circuits (ASICs) that are preprogrammed and designed to route between Ethernet ports at high speed. A traditional router is connected to the L3 switch and handles the routing of any traffic that needs to be sent across the WAN. Because the L3 switch does not need the flexibility required of the router to support different WAN protocols, it can use ASICs to route traffic at the 100-Mbps speeds expected of the LAN network. The router in the network is designed to handle the requirements of routing at T1 (1.5 Mbps) speeds and would cause a bottleneck if it had to route between VLANs, as routing is performed in software, not hardware. Of course, you could purchase an expensive high-performance router with three Ethernet ports and a T1 interface; however, the cost associated with this approach is much higher. The cost associated with adding more routed Ethernet ports to the router (e.g., if a new VLAN was added to the network) is also high.

Layer 3 Routing Versus Layer 3 Switching

It is important to understand the difference between Layer 3 routing and Layer 3 switching. Both terms are open to some interpretation; however, the distinction between both can perhaps be best explained by examining how an IP packet is routed. The process of routing an IP packet can be divided into two distinct processes:

- **Control plane—**The control plane process is responsible for building and maintaining the IP routing table, which defines where an IP packet should be routed to based upon the destination address of the packet, which is defined in terms of a next hop IP address and the egress interface that the next hop is reachable from. Layer 3 routing generally refers to control plane operations.
- **Data plane—**The data plane process is responsible for actually routing an IP packet, based upon information learned by the control plane. Whereas the control plane defines where an IP packet should be routed to, the data plane defines exactly how an IP packet should be routed. This information includes the underlying Layer 2 addressing required for the IP packet so that it reaches the next hop destination, as well as other operations required on for IP routing, such as decrementing the time-to-live (TTL) field and recomputing the IP header checksum. Layer 3 switching generally refers to data plane operations.

Figure 6-2 illustrates the differences between control plane operation and data plane operation by providing an example of how an IP packet is routed.



NOTE

Some Cisco Catalyst Layer 3 switches support the Layer 3 switching of Internetwork Packet Exchange (IPX) packets as well. For this chapter, the discussion focuses purely on IP packets.

Figure 6-2 Control Plane and Data Plane Operation

In Figure 6-2, Host-A is sending an IP packet to Host-B over a LAN network that includes a couple of routers. The following describes the events that occur in Figure 6-2.

Step 1 Host-A (1.1.1.10) needs to send an IP packet to Host B (3.3.3.10). Host-A determines (by considering its own IP address, its subnet mask, and the IP address of Host-B) that Host-B is a non-local host and, therefore, must send the IP packet to the configured default gateway of 1.1.1.1 (Router-A). Because Host-A is connected to the network via Ethernet, Host-A must deliver the original IP packet in an Ethernet frame to Router-A. To place the packet in an Ethernet frame that can be delivered to Router-A, Host-A must know the MAC address of Router-A's 1.1.1.1 interface. Host-A checks the local Address Resolution Protocol (ARP) cache to see whether or not it knows the MAC address of Router-A (1.1.1.1). Assuming Host-A does not know the MAC address, Host-A broadcasts an ARP request, which is sent to all devices on the local LAN and asks for the MAC address associated with the IP address 1.1.1.1.

Step 2 Because Router-A is configured with an IP address of 1.1.1.1 on the interface attached to Host-A, it responds to the ARP request by sending a unicast ARP reply, which provides its MAC address (0000.0001.0001).

Step 3 Host-A can now encapsulate the IP packet in an Ethernet frame and send it to Router-A. The destination MAC address of the frame is the MAC address of Router-A, which ensures that Router-A receives the IP packet contained within for routing. The destination IP address, however, is not that of Router-A; it's that of Host-B, the true eventual destination of the packet (in other words, the IP addresses in the packet are not modified).

Step 4 Router-A receives the Ethernet frame and the data plane operations begin. For Router-A to forward the packet on to the appropriate next hop, it must know who the next hop is and the MAC address of the next hop. To determine the next hop, the router inspects the destination IP address of the IP packet (IP routing is always based upon the destination IP address). Router-A references the local route table for an entry that matches the destination IP address (3.3.3.10) and finds that 3.3.3.0/24 is reachable via a next hop IP address of 2.2.2.2 (Router-B).

Step 5 Because Router-A is connected to Router-B via Ethernet, Router-A must send the IP packet inside an Ethernet frame addressed to Router-B. To determine the MAC address associated with the next hop router, the local ARP cache on the router is checked to see if an entry exists for the IP address of the next hop. If no entry exists, then the router must generate an ARP request, asking for the MAC address associated with the next hop IP address (this is a control plane operation). Once the correct destination MAC address is known, the routed frame destination MAC address can be rewritten. The source MAC address is also rewritten to the MAC address of the Ethernet 1 interface on Router-A so that Router-B knows it received the frame from Router-A. It is this process of rewriting the frame MAC addresses that represents the key concept of data plane operations—A router does not modify the source or destination IP addresses of IP packets that are being

delivered, but rather it must *rewrite* the destination and source MAC address so that the IP packet can be delivered over the LAN to the next hop.

NOTE

Router-A actually does have to modify some information in the IP header. Router-A must decrement the IP time-to-live (TTL) field and also must recompute the IP header checksum, since the TTL field has been changed. IP addressing might also be modified if network address translation (NAT) is configured; however, this operation is performed by a separate process outside of the control plane and data plane operations of routing.

Step 6 The rewritten Ethernet frame containing the IP packet is sent to Router-B.

Step 7 Router-B receives the frame from Router-A and examines the destination IP address of the packet. Because the destination IP address is that of a host that is locally connected, Router-B can complete the delivery by sending the packet to Host-B. Because Host-B is connected via Ethernet to Router-B, Router-B must send the IP packet inside an Ethernet frame addressed to Host-B. The same rewrite of the destination (and source) MAC address that was described in Step 5 takes place, and the frame is delivered to its final destination, Host-B.

NOTE

It is important to understand that the MAC addresses are specific only to each local LAN. For example, Host-A does not know and does not need to know Host-B's MAC address or even Router-B's MAC address. Host-A needs to know only the MAC address of Router-A so that it can deliver IP packets in Ethernet frames locally to Router-A, with Router-A then forwarding the packet on appropriately and with this process occurring on a hop-by-hop basis until the final destination is reached.

Control Plane and Data Plane Implementation

Control plane operations require an understanding of routing protocols and hence require some intelligence that is capable of supporting the complex algorithms and data structures associated with protocols such as Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP). Depending on the routing protocol(s) configured, the control plane operations required might vary dramatically between different routing devices. On the other hand, data plane operations are simple and fixed in their implementation because how a packet is routed is the same, regardless of the routing protocol that was used to learn where a packet should be routed. Although data plane operations are simple, they are also performed much more frequently than control plane operations because data plane operations must be performed for every packet that is routed, while control plane operations must be performed only for routing topology changes once the routing table is built. This means that the performance of the data plane implementation ultimately dictates how fast a routing device can route packets.

Because control plane operations are complex, most vendors use a general purpose CPU capable of supporting a high-level programming language so that vendors can easily develop and maintain the complex code associated with support the various routing protocols. In this respect, the control plane is implemented in *software,* which means that code (software) developed from a high-level programming language provides control plane operation. Both traditional routers and Layer 3 switches normally take the same approach to implementing the control plane operations associated with IP routing, using software that requires a general purpose CPU.

In contrast to control plane operations, data plane operations are very simple. In fact, the data plane operations required can be presented in a single table. Table 6-1 describes the data plane operations that

must take place, assuming a packet is addressed from a host called Host-A to another host called Host-B and is sent via a router.

Table 6-1 Data Plane Operations Required on Received Frames

	Layer 2 Ethernet Header		Layer 3 IP Header				Data	FCS
	Destination MAC	Source MAC	Destination IP	Source IP	TTL	Checksum		
Received Frame	Router MAC Address	Host-A MAC Address	Host-B	Host-A	n	value1		
Rewritten Frame	Next Hop MAC Address	Router MAC Address	Host-B	Host-A	n-1	value2		

In Table 6-1, the details of the received frame are indicated and then the details required for the rewritten frame that is transmitted after routing are shown. Notice that the following fields must be modified for the rewritten frame that is forwarded to the next hop routing device:

- **Destination MAC address—**The MAC address of the next hop must be written to the rewritten frame.
- **Source MAC address—**The source MAC address must be written to the MAC address of the router.
- **IP TTL—**This must be decremented by one, as per the normal rules of IP routing.
- **IP Header Checksum—**This must be recalculated, as the TTL field changes.

The process of how the data plane operations shown in Table 6-1 are implemented is where the difference between a traditional router and Layer 3 switch lie. A traditional router uses the same general purpose CPU used to perform control plane operations to also implement data plane operations, meaning data plane operations are handled in software. A Layer 3 switch on the other hand uses an ASIC to perform data plane operations because it is very easy to program the very simple operations required for the data plane into an ASIC. In this respect, the data plane is implemented in hardware because a series of hardware operations are programmed into the ASIC that perform the data plane operations required for routing a packet.

NOTE

It should be noted that many high-end routers use ASICs for data plane operations in a similar fashion to Layer 3 switches. In fact, much of the ASIC technology used in Layer 3 switches is derived from the ASICs used in high-end routers.

So how does this affect performance? Well, a general purpose CPU is designed to support many different functions, where as an ASIC is designed to support a single function or a handful of specific functions such as performing the data plane operations required to route a packet. This means that an ASIC can operate much faster because the internal architecture of the ASIC can be optimized just to perform the operations required for data plane operations, whereas a general purpose CPU must be designed to support a series of generic functions that do not relate to data plane operations whatsoever (as the CPU must support other applications). A high-level language combines the generic functions of the general purpose CPU to provide the higher specific functions required to perform data plane operations. This approach allows flexibility but comes at the price of performance. Hence, a Layer 3 switch that performs data plane operations using ASICs route packets much faster than a traditional router that performs data plane operations using a general purpose CPU.

NOTE

The term *software* when applied to Layer 3 routing means that a general purpose CPU performs routing, along with other tasks such as system maintenance and providing command-line access. The term *hardware* when applied to Layer 3 switching means an ASIC dedicated to the process of Layer 3 switching, whose sole purpose in life is to route packets.

Hardware-Based Layer 3 Switching Architectures

Although the data plane operations required for routing IP packets can easily be accelerated by the use of ASICs, it is important to understand that a fundamental requirement for data plane operation is the process of determining the next hop IP address for the destination IP address of the packet and the MAC address associated with the next hop so that the correct destination MAC address can be written to the rewritten frame. The components that implement data plane operations must "look up" this information (see the lookup operation in Figure 6-2); this lookup operation in itself can become a bottleneck. To ensure the lookup process does not significantly delay the rewrite processes of data plane operation, Layer 3 switches use specialized data structures that allow for fast lookups. These data structures can be split into two categories:

- **Route cache—**A route cache is populated with information that defines how to Layer 3 switch frames associated with a particular *flow.* A flow uniquely identifies specific traffic conversations in the network (e.g., one flow might be Host-A communicating with Host-B, while another flow might be Host-A communicating with Host-C), and each flow entry contains the required information to Layer 3 switch packets received for that flow. The flow entries are built by routing the first packet in software, with the relevant values in the rewritten first frame used to fill out the required information for a flow entry. Subsequent packets associated with the flow are then Layer 3 switched in hardware based upon the information learned in the flow entry. Cisco's implementation of route caching on Cisco Catalyst switches is called *Multilayer switching* (*MLS*), and is discussed in more detail in Scenario 6-1.
- **Optimized route lookup table—**One approach to the lookup process could be to use the routing table; however, this contains information not relevant to data plane operations, such as the routing protocol that learned a route, metric associated with a route, and the administrative distance of a route. The routing table also does not contain MAC address information for the next hop. This must be determined either via a control plane operation (using ARP) or by reading the ARP cache. Next-generation Cisco Catalyst Layer 3 switches use an optimized route lookup table, which organizes only the required routing information for data plane operations (e.g., destination prefix, next hop, egress interface) and also includes a pointer to another optimized adjacency table, which describes the MAC address associated with the various next hop devices in the network. Cisco's implementation of using optimized route lookup tables on Cisco Catalyst switches is called *Cisco Express Forwarding* (*CEF*) and is discussed in more detail in Scenario 6-2 and Scenario 6-4.

It is important to note that in addition to possessing a high performance lookup mechanism, many Layer 3 switches also possess specialized hardware that can be used to provide QoS classification and security access control (using access control lists) for packets at the same time the next hop lookup is being implemented. This means that these features can be turned on with affecting performance.

Cisco Catalyst 6000/6500 Switch Architecture

The Catalyst 6000/6500 is the flagship of the Cisco Catalyst switching family and represents one of the most popular switches used for enterprise networks and service providers. If you are tasked with the procuring a Catalyst 6000/6500 switch, it is important to understand the various Supervisor modules available and the technologies that are used with each to perform both Layer 2 and Layer 3 switching.

The following topics are now discussed:

- Supervisor architectures
- Catalyst 6000/6500 operating systems

Supervisor Architectures

Several architectural options are available when designing a Catalyst 6000/6500 switch, each of which varies in terms of Layer 3 capabilities. Layer 3 capabilities are added to the Catalyst 6000/6500 switch by two key components:

- **Policy feature card (PFC)—**The PFC provides the necessary ASICs to perform hardware-based Layer 3 switching, quality of service (QoS) classification, and access control list (ACL) filtering. The PFC requires a *route processor* to populate the route cache or optimized route table structure used by the L3 switching ASIC. If no route processor is present, the PFC can perform only Layer 3/4 QoS classification and ACL filtering and cannot perform L3 switching.
- **Multilayer switching feature card (MSFC)—**The MSFC is essentially a Cisco IOS router based upon the high-performance 7200 series router. This provides the route processor functions required by the PFC to implement L3 switching. The MSFC provides the necessary routing information in the PFC route cache so that the PFC can L3 switch packets.

When you purchase the Catalyst 6000/6500, you have a choice as to the generation of Supervisor that you wish to purchase, as well as the L3 components (i.e., the PFC and/or MSFC) that you require depending on your Layer 3 requirements. These options include the following:

- Supervisor 1A
- Supervisor 2
- Supervisor 720—Next-generation Supervisor engine that includes an integrated PFC-3, MSFC-3, and 720-Gbps crossbar switching matrix.

Each of the various configuration options is now examined.

Supervisor 1A with no PFC

The simplest configuration option available for the Catalyst 6000 is just the Supervisor 1 module with no policy feature card (PFC) or MSFC. In this configuration, the switch is essentially a Layer 2 switch and possesses no Layer 3 switching or classification capabilities. A Supervisor 1A can provide a Layer 2 switch up to 15 million packets per second (Mpps).

Supervisor 1A with PFC-1

The next option available for the Catalyst 6000/6500 is the Supervisor 1 module with a policy feature card (PFC-1) installed. The PFC-1 enables Layer 3 and 4 classification for QoS classification and security ACL filtering; however, L3 switching is not supported unless an MSFC is added to provide route processor functions. The Supervisor 1A with PFC-1 is capable of processing frames through the QoS and ACL engines without degrading Layer 2 switching performance, at speeds of up to 15 Mpps. Figure 6-3 demonstrates the architecture of the Supervisor 1A with PFC-1.

Figure 6-3 Supervisor 1A with PFC-1 Architecture

In Figure 6-3, the Supervisor 1A contains the basic Layer 2 engine that references the local bridge table for determining the egress port for switching decisions. The PFC contains a Layer 3 engine, flow cache, ACL engine, and ACL table. In this configuration, the PFC is not used for L3 switching**,** because no route processor (provided by an MSFC) is installed that provides the required next hop information. However, the PFC can be used for Layer 3/4 QoS classification and ACL filtering; the ACL engine is responsible for providing these functions. The ACL table is stored in *ternary content addressable memory (TCAM)*, which stores ACL information in a format that can be referenced very quickly by the ACL engine. When a packet arrives that requires ACL filtering, while the L2 engine determines the forwarding decision to be made based upon the information contained within the L2 bridge table, at the same time, the ACL engine determines whether or not the packet is permitted or denied. Because the L2 lookup and ACL lookup occur in parallel, applying ACLs or QoS classification to traffic does not affect the forwarding rate of the switch (15 Mpps).

Supervisor 1A with PFC-1 and MSFC-1/MSFC-2

The last Supervisor 1A option and only L3 switching option for the Catalyst 6000/6500 using the Supervisor 1A is the Supervisor 1A module with PFC-1 and MSFC-1 or MSFC-2 installed. The MSFC-1 is now end of sale, so you can only purchase the MSFC-2 if you want to add Layer 3 switching capabilities to existing Supervisor 1A configurations.

NOTE

The MSFC-1 and MSFC-2 differ only in performance. The MSFC-1 has an R5000 200-MHz processor, supports up to 128MB memory, and can route packets at up to 170 Kpps in software. The MSFC-2 has an R7000 300-MHz processor, supports up to 512 MB memory, and can route packets at up to 650 Kpps in software. The Layer 3 switching performance in hardware is still 15 Mpps, regardless of the MSFC used.

In this architecture, the L3 engine onboard the PFC-1 can perform L3 switching, because a route processor is now present in the form of the MSFC. Figure 6-4 shows the architecture of the Supervisor 1A with PFC-1 and MSFC.

Figure 6-4 Supervisor 1A with PFC-1 and MSFC

In Figure 6-4, the addition of the MSFC allows for the L3 engine to L3 switch inter-VLAN traffic. All other features of the PFC, such as QoS classification and ACL filtering are also supported. The PFC-1 and MSFC-1/MSFC-2 use *multilayer switching (MLS)* to perform L3 switching; this means that a flow cache exists on the PFC which is used to L3 switch packet flows through the switch. The first packet within a flow must always be routed by the MSFC, which references the routing table to determine the next hop information for a packet. Once the MSFC has made a routing decision and forwarded the frame back to the L3 engine, the L3 engine reads the routed frame information and writes this information into the flow cache. Subsequent packets received and that match flow cache entries can now be L3 switched by the L3 engine, rather than the MSFC. A limitation of the MLS L3 switching mechanism is the initial route lookup performed in software by the MSFC. The first packet in an IP flow must be passed to the MSFC route processor for routing. In an environment that has many connections being established at the same time, this can cause performance problems for the MSFC. This problem in particular applies to service provider environments, which typically must handle conditions where many short term connections (e.g., downloading a web page might open several HTTP connections that are terminated immediately once the page is downloaded) are being established at once. The Supervisor 1 with PFC-1 and MSFC can L3 switch packets at 15 Mpps.

Supervisor 2 with PFC-2

The first configuration available for the Catalyst 6000/6500 with a Supervisor 2 module is the Supervisor 2 with a policy feature card 2 (PFC-2) installed (the Supervisor 2 is integrated with PFC-2; you can't purchase either separately). The PFC-2 is similar in function to the PFC-1, enabling Layer 3 classification for QoS classification and security ACL filtering; however, it is twice as fast as the PFC-1 and supports more ACLs that can be stored in hardware for QoS and Security. The Supervisor 2 with PFC-2 is capable of switching packets and performing Layer 3/4 QoS classification and ACL filtering at up to 30 Mpps; however, this requires switch fabric enabled modules and a switch fabric module to be installed. Because no MSFC is present in this configuration, L3 switching is not possible. Figure 6-5 demonstrates the architecture of the Supervisor 2 with PFC-2.

Comparing the architecture of the Supervisor 1A with PFC-1, notice that the PFC-2 is actually an integrated part of the Supervisor 2 module. The most notable difference is that the Layer 2 and ACL engine are now combined into a single L2/L4 engine, which boosts the performance capabilities of L2 switching combined with Layer 3/4 QoS classification and ACL filtering up to 30 Mpps. The L3 engine is not used for L3 switching, because an MSFC-2 (route processor) is required to generate information contained in the CEF table.

Figure 6-5 Supervisor 2 with PFC-2 Architecture

Supervisor 2 with PFC-2 and MSFC-2

To enable Layer 3 switching on a Supervisor 2 with PFC-2, the only option is to add an MSFC-2 (the MSFC-1 is not supported on the Supervisor 2). In this architecture, the L3 engine onboard the PFC-2 can perform L3 switching, because a route processor is now present in the form of the MSFC-2. Figure 6-6 shows the architecture of the Supervisor 2 with PFC-2 and MSFC-2.

In Figure 6-6, the addition of the MSFC allows for the L3 engine to L3 switch inter-VLAN traffic. All other features of the PFC, such as QoS classification and ACL filtering, are also supported. The PFC-2 and MSFC-2 use CEF to perform L3 switching; the MSFC-2 is responsible for generating the appropriate CEF tables (the FIB table and adjacency table, discussed in Scenario 6-2) upon PFC initialization. This means that as soon as packets need to be L3 switched, the L3 engine has the necessary information to L3 switch the packet, without having to send the first packet associated with a flow to the MSFC (as is the case with MLS). This architecture eliminates the issue that MLS has for supporting an environment that has thousands of connections being established every second. The Supervisor 2 with PFC-2 and MSFC-2 can L3 switch packets at 30 million packets per second.

Figure 6-6 Supervisor 2 with PFC-2 and MSFC-2

The Switch Fabric Module

The *switch fabric module (SFM)* is a module that includes a switching backplane that increases the forwarding rate of the Catalyst 6500 backplane from 32 Gbps to 256 Gbps.

NOTE

The SFM is available only for the Catalyst 6500 with Supervisor 2 and must be installed in Slot 5. A redundant SFM is available and must be installed in Slot 6 if used.

The SFM provides a crossbar switching matrix for the switching backplane, which allows multiple frames to be switched between different line cards at the same time. For example, a frame can be switched across the matrix from line card #2 to line card #4 at exactly the same time as another frame is being switched from

line card #3 to line card #8. This is not possible on the traditional shared 32-Gbps backplane of the Catalyst 6000/6500; thus the crossbar matrix can support much higher packet forwarding rates. The SFM provides 16 * 8-Gbps full-duplex connections into the switching matrix. Figure 6-7 shows the SFM and how it provides connections to the other switch modules in the switch chassis.

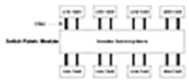


Figure 6-7 The Switch Fabric Module

Each switch module has two available 8-Gbps connections to the SFM. Depending on the type of line cards installed, a line card might take advantage of zero, one, or both of the 8-Gbps connections. Three types of switch modules (line cards) relate to the SFM:

- **Non fabric-enabled card—**These cards are not compatible with the SFM and connect only to the 32-Gbps backplane.
- **Fabric-enabled card—**These cards are compatible with both the SFM and 32-Gbps backplane. A single 8-Gbps connection is provided to the SFM, as well as a single connection to the traditional 32-Gbps backplane.
- **Fabric-only card—**These cards connect solely to the SFM via dual 8-Gbps connections. These cards do not connect directly to the traditional 32-Gbps backplane.

It is important to understand that all of the cards listed above can communicate with each other. The fabric-only card can communicate with non fabric-enabled cards because the SFM has a connection to the traditional Catalyst 6000/6500 32-Gbps backplane.

The Distributed Feature Card (DFC)

The *Distributed Feature Card (DFC)* allows fabric-enabled line cards to make L3 forwarding decisions locally without requiring the L3 switching engine located on the Supervisor PFC. The DFC consists of the same components as the PFC located on the Supervisor module, however it does not contain the MSFC routing engine. Figure 6-8 shows the DFC architecture.

NOTE

Only fabric-enabled line cards support the DFC. If you are using the DFC, you must install a switch fabric module card.

Figure 6-8 The Distributed Feature Card

In Figure 6-8, you can see a fabric-enabled line card that has a DFC installed. The DFC looks exactly like a PFC and performs the same functions as the PFC, except for frames received on local ports. The key to the DFC is the use of distributed CEF (dCEF). A master CEF table resides on the Supervisor 2 PFC-2, which is generated by the MSFC routing table. The master CEF table is downloaded (mirrored) to each DFC, which enables the L3 engine on each DFC to make routing decisions locally. If a route table change occurs, the CEF tables on the PFC and each DFC are updated immediately. If frames are received on a DFC-enabled line card that require routing, the L3 engine on the DFC inspects the destination IP address of the IP packet contained within the frame and looks up the CEF table to determine the next-hop MAC address and egress port. If the egress port is local, the L3 engine rewrites the destination MAC address and forwards the frame out the appropriate local egress port. If the egress port is located on another module, the L3 engine rewrites the destination MAC address and forwards the frame onto the SFM matrix, prepending a tag that identifies the egress port the frame should be switched out of. The tagged frame is forwarded to the appropriate switch module, with the local switching engine forwarding the frame out the appropriate egress port. This forwarding of frames across the SFM matrix does not require any intervention by the main Supervisor 2 PFC L3 engine. Given that a DFC can L3 switch up to 30 Mpps, if a Catalyst 6509 has a single Supervisor 2 with PFC-2 and MSFC-2, a SFM, and seven fabric-

enabled line cards each with a DFC installed, the total system capacity theoretically is 210 Mpps (7 * 30 Mpps).

Supervisor 720

A recent new addition to the Catalyst 6500 family is the Supervisor 720 engine, which is the third-generation supervisor engine that integrates the following components into a single module:

- PFC-3
- MSFC-3
- Crossbar Switching Fabric that provides 720 Gbps of backplane bandwidth

The Supervisor 720 significantly increases the number of slots available for data modules. For example, in a non-redundant Catalyst 6509 configuration, the Supervisor 720 takes up only a single slot, leaving eight slots for data modules. In comparison, a Supervisor 2 with SFM installed takes up two slots, leaving only seven slots for data modules. In a redundant configuration, the Supervisor 720 engines take up only two slots while the Supervisor 2 engines and redundant SFMs take up four slots.

NOTE

For the Catalyst 6506, 6509, and 6513, the primary Supervisor 720 must be installed in Slot 5, while the redundant Supervisor 720 must be installed in Slot 6. You must also install a minimum of 2500W power supplies to power the Supervisor 720 on all Catalyst 6500 switches.

The Supervisor 720 also provides a large number of feature enhancements, which include the following:

- Hardware-based MPLS forwarding
- Hardware-based IPv6 Layer 3 switching
- Support for hardware assisted NAT and generic routing encapsulation (GRE)
- Backplane bandwidth increases to 2 * 20 Gbps, up from 2 * 8 Gbps with the SFM
- Maximum throughput of 400 Mpps, almost twice that of the Supervisor 2 with SFM

Catalyst 6000/6500 Operating Systems

On the Catalyst 6000/6500, it is important to understand that the switch can operate in one of three different modes, depending on the hardware installed and operating systems used to manage the switch. These modes include the following:

- **CatOS**—In this mode, the switch only operates a single operating system: CatOS. No MSFC is installed, because this uses its own operating system.
- **Hybrid mode**—Hybrid mode refers to the configuration where an MSFC is installed that is running Cisco IOS, whilst the switch is running CatOS. This means two separate management interfaces are required—one for the switch and one for the MSFC.
- **Native mode**—In this configuration, a single Cisco IOS operating system is used to manage both the switch and the MSFC. This allows for a single management interface to manage both the switching and routing components of the switch (native mode requires an MSFC).

Scenario 6-1: Configuring MLS on the Catalyst 6000

In this scenario, you learn how to configure *Multilayer switching* (*MLS*), which represents the first Layer 3 switching technology used by Cisco Catalyst switches to provide wire-speed routing of inter-VLAN traffic. Although MLS is no longer considered the ideal L3 switching architecture, many installations still use MLS, and hence, you as a Cisco engineer must be able to configure and troubleshoot MLS.

Figure 6-9 illustrates the topology used for this scenario. In Figure 6-9, an MLS configuration is to be used to provide the high-speed Layer 3 switching of inter-VLAN traffic on Switch-A. Router-A is required to provide routing control plane operation, initially routing the first packet of each flow sent through the Switch-A, allowing Switch-A to learn the required MAC address rewrite operations for Layer 3 switching.

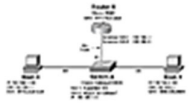


Figure 6-9 Scenario 6-1 Topology

The following describes the function of each component of the lab topology shown in Figure 6-9:

- Switch-A is a Catalyst 5509 switch with a Supervisor 3G module installed. In this scenario, Switch-A acts as the *MLS Switching Engine (MLS-SE)*, which is responsible for the data plane operations required for Layer 3 switching.
- Router-A is a Cisco 3620 router with a physical FastEthernet interface that connects as an 802.1Q to Switch-A. Two virtual interfaces are required on the trunk to provide inter-VLAN routing between each VLAN. Router-A acts as an *MLS Route Processor (MLS-RP)*, which is responsible for making routing decisions for the first packet associated with an MLS flow.
- Host-X and Host-Y are workstations that are used to test that inter-VLAN routing works with MLS configured.

Understanding MLS

MLS represents the first hardware-based Layer 3 switching mechanism used by Cisco Catalyst switches, supported on switches such as the Catalyst 5000/5500 and Catalyst 6000/6500. This section explains the operation of MLS and how Layer 3 switches use MLS.

MLS Overview

MLS is designed to support a distributed L3 switching architecture, which means the various components of MLS do not need to be located on the same physical device. MLS consists of the following two main components:

- **MLS Route Processor (MLS-RP)—**This component represents the *control plane* of the routing process. The MLS-RP maintains the route table and is responsible for updating the route table as changes in the network topology occur.
- **MLS Switching Engine (MLS-SE)—**This component represents the *data plane* of the routing process. The MLS-SE is responsible for determining the next hop and egress interface information for each frame received that requires routing, and then rewriting the frame as required and forwarding the frame to the correct egress interface.

The route table is maintained on the MLS-RP, and this control plane information must somehow be communicated to the MLS-SE. The MLS-RP and MLS-SE communicate using the *MLS protocol (MLSP)*, which is a Cisco proprietary protocol that uses multicast Ethernet frames to communicate.

Flows are used to represent routing information in the route cache located on the MLS-SE. A flow can be defined based upon a unique destination IP address, a unique combination of source and destination IP address, or a unique combination of source and destination IP address, as well as source and destination Layer 4 (i.e., TCP or UDP) ports. For example, all packets that are sent from any source IP address to a destination IP address of 192.168.2.1 can be represented as a flow. All packets sent from a source IP address of 192.168.2.1 to a destination IP address of 192.168.2.2 can be represented by a flow, with the return packets represented as another flow.

For each flow, the MLS-SE builds the required frame rewrite information for Layer 3 switching (i.e., source and destination MAC address rewrite information) by allowing the MLS-RP to perform the normal routing process for the first frame of each new flow. This allows the MLS-SE to learn the required rewrites for the source and destination MAC address of a framed IP packet after it has been routed, with the appropriate information stored in an MLS cache. Figure 6-10 demonstrates the MLS architecture and how packets are routed using MLS using flows.

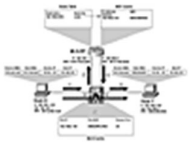


Figure 6-10 MLS Architecture

The following describes the events that occur in Figure 6-10:

Step 1 Host-X sends an IP packet to Host-Y (192.168.2.100). Because the packet is on a different IP subnet from Host-X, Host-X addresses the Ethernet frame containing the IP packet to the configured default gateway (MLS-RP) for routing, meaning the frame has a destination MAC address of 0010.7be2.aba0.

Step 2 The frame is received by the MLS-SE, which initially examines the destination MAC address of the frame. Because the destination MAC address is the MAC address of the MLS-RP (0010.7be2.aba0), the MLS-SE immediately marks the frame as a *candidate frame* for L3 switching (as any L3 switched frame always has a destination MAC address of a routing device). The MLS-SE inspects the destination IP address in the packet and checks the MLS cache for a flow. Because this is the first packet sent to Host-Y, no entry is present so the packet is sent to the MLS-RP for routing. An incomplete flow entry is written in the MLS cache, which includes only information that identifies the flow at this stage (e.g., destination IP address).

Step 3 The MLS-RP receives the frame and performs normal IP routing, inspecting the destination IP address and determining from the local route table that the destination is locally attached. The MLS-RP determines the MAC address of Host-Y by checking the ARP cache (and sending an ARP request if the ARP cache does not contain the entry) and generates a new Ethernet frame to transport the IP packet to its intended destination.

Step 4 The MLS-SE receives the routed frame and writes the destination MAC address of the routed frame into the incomplete flow entry that was initially created in Step 2. The switch also consults the local bridge table to determine the egress port associated with the destination MAC address and writes this

information into the flow entry as well. The flow entry information is used for future rewriting and forwarding of packets sent to Host-Y without having to forward the packets to the MLS-RP.

Step 5 The frame is switched out the appropriate egress port (2/3) to Host-Y.

Step 6 Host-X sends another IP packet to Host-Y. The MLS-SE sees from the destination MAC address that the frame is destined to the MLS-RP and is therefore a candidate for L3 switching. The MLS-SE inspects the destination IP address (192.168.1.200) and matches it against the flow entry completed in Step 4. The MLS-SE rewrites the destination MAC address of the frame and performs other necessary L3 switching operations (such as decrementing the IP TTL and computing the IP and Ethernet checksums).

Step 7 The rewritten frame is switched out the correct egress port (2/3) to Host-Y. All subsequent IP packets from Host-X to Host-Y are L3 switched as described in Step 6 and Step 7 as long as the flow entry created in Step 4 is valid.

As you can see from Figure 6-10, MLS requires the first packet in a flow to be routed through the MLS-RP, which allows the MLS-SE to determine the appropriate information that must be rewritten by the MLS-SE for the subsequent L3 switching of packets to Host-Y.

The MLS-RP and MLS-SE also communicate regularly, so that if the MLS-SE can detect if an MLS-RP goes down, the MLS-SE can flush the appropriate flow entries in the MLS cache. This is important in a redundant topology where two or more MLS-RPs provide inter-VLAN routing, because it ensures the redundant MLS-RP can be used if the primary MLS-RP fails.

MLS Flows

Because the MLS architecture is based on flows, it is important to understand the different type of flows that exist. Each flow can be categorized as belonging to a particular flow mask. A flow in MLS terms represents one of the following (the name of the flow mask that categorizes the flow is indicated in parentheses):

- **Destination IP address (destination-only)—**A single destination IP address can represent a flow. All traffic sent to the destination IP address is part of a single flow. In Figure 6-10, flows were represented by destination IP address.
- **Source and Destination IP address (source-destination)—**All traffic sent between a specific source and destination IP address is part of a single flow. This means that several flows might exist for the same destination IP address; each flow is differentiated by the source IP address.
- **Full flow (full)—**A full flow represents all traffic associated with a specific source IP address, destination IP address, source TCP/UDP port, and destination TCP/UDP port. For example, a Telnet connection between 192.168.1.1 and 192.168.2.1 would be represented by a separate flow from an HTTP connection between the same two hosts.

Flows are important because the route cache used by MLS uses flows to store the information for the hardware-based rewrites required for Layer 3 switching process. The flow you use depends on the requirements of your network. For example, if you are using simple routing (where routing decisions only need to be made based solely upon the destination IP address of each packet), destination IP address flows are only required. However, if you are using access control lists (ACLs) on a routed interface through which a packet would normally travel, you need to use source and destination IP address or full flows depending on the granularity of the ACLs. For example, Figure 6-11 shows the topology of Figure 6-10 with an extended ACL configured on an interface on the MLS-RP.

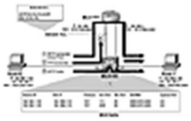


Figure 6-11 MLS Architecture and ACLs

The following describes the events that occur in Figure 6-11:

Step 1 Host-X attempts to establish an HTTP connection to Host-Y, sending an IP packet with a source IP address of 192.168.1.100, destination IP address of 192.168.2.100, protocol number of 6 (TCP), destination TCP port of 80, and a random source TCP port of 1111. The frame for the IP packet is addressed to the MLS-RP, because Host-Y is not on the local subnet. The MLS-SE receives the frame, marks it as a candidate frame, and routes the frame to the MLS-RP, because no flow entry exists that matches the packet. When the MLS-RP receives the packet, it is inspected against the ACL and is permitted because the packet is a TCP packet with a destination port of 80. The MLS-SE receives the routed packet, allowing it to write a complete flow entry in the MLS cache.

Step 2 Host-X next attempts to establish an FTP connection to Host-Y. Because the flow cache on the MLS-SE is now using a full flow mask, the FTP packet sent does not match the flow entry created in Step 1 and hence is forwarded to the MLS-RP. The MLS-RP inspects the packet against the ACL and drops the packet because it is not an HTTP packet. This means that the MLS-SE never sees the return routed frame come back from the MLS-RP that includes the required information to complete the entry for the flow, meaning the MLS-SE can never complete the incomplete flow entry created when the packet was first received by the MLS-SE. Any subsequent FTP connection requests (or any other non-HTTP traffic) is always forwarded to the MLS-RP (because no complete flow entries ever exist on the MLS-SE), at which point the traffic is dropped.

Step 3 All traffic associated with the HTTP connection established in Step 1 is Layer 3 switched by the MLS-SE. If a new HTTP connection is established between Host-X and Host-Y, a new flow entry must be built (as per Step 1), because the source TCP port is different for each connection. In Figure 6-11, you can see two flow entries, which each represent separate HTTP connections established from Host-X to Host-Y.

In Figure 6-11, a full flow *must* be used because the MLS-RP must be able to permit traffic based on specific combinations of source and destination IP address and source and destination TCP/UDP ports. If a destination or destination-source flow were used in Figure 6-11, the MLS-SE would not differentiate HTTP packets from FTP packets and FTP packets would be incorrectly L3 switched (permitted) to Host-Y.

NOTE

The MLS-RP can communicate the required flow mask to the MLS-SE, which ensures that if an ACL is applied in an existing MLS configuration, the appropriate flow mask is immediately used on the MLS-SE ensuring packets are not incorrectly permitted if they have been denied in the ACL.

MLS Communications

MLS requires some internal communications between the MLS-RP and MLS-SE to ensure L3 switching on the MLS-SE is performed accurately and correctly. Two major events that can cause the MLS cache to become invalid:

- Routing topology changes
- ACL configuration

If a routing topology change occurs, it is possible that flow information in the MLS cache needs to be updated. This is also the case if an ACL is applied to an interface (or modified) on the MLS-RP. The MLS-RP and MLS-SE use MLSP messages to communicate with each other. The MLS-RP sends MLSP messages to the MLS-SE if either of the above events occur, which indicate to the MLS-SE that it should flush the MLS cache and possibly modify the flow mask used. An example of when the flow mask would be changed is when an extended ACL is applied to a previously unfiltered interface. As you saw in Figure 6-11, a full flow mask is required when extended ACLs are used, so the MLS-RP sends an MLSP message to the MLS-SE to flush its MLS cache and update the flow mask.

NOTE

MLSP communications are also used to verify that MLS components are still alive via the exchange of hello packets. These messages are sent every 15 seconds by default.

In an environment where multiple MLS-RPs are present, the MLS-SE must be able to differentiate between each MLS-RP. This can be done based upon the MAC address of each MLS-RP; however, if there are thousands of flow entries in the MLS cache and if each of the entries associated with an MLS-RP that has just gone down need to be flushed, searching through the cache based upon a 48-bit MAC address value can take some time. To facilitate faster cache purges, an 8-bit XTAG value is assigned to MLS-RP, which acts like an index for each MLS-RP, allowing the MLS-SE to differentiate between the flow entries associated with each MLS-RP much more quickly.

Cisco Platform Support for MLS

It is important to understand that MLS is now considered a legacy technology and as such is not supported on many newer Cisco Catalyst switches. Table 6-2 lists the Cisco Catalyst switch platforms that the MLS-SE component is supported on.

Table 6-2 MLS-SE Supported Platforms

Platform	Hardware Requirements	Software Requirements
Catalyst 5000/5500	Supervisor 2G/3G/3/3F NetFlow Feature Card	CatOS 4.1(1)
Catalyst 6000/6500[1]	Policy Feature Card	5.1(1)

[1]The Catalyst 6000/6500 MLS-SE supports only operation with a locally installed MSFC as the MLS-RP. No support for an external MLS-RP is currently provided.

Table 6-3 lists the router platforms that the MLS-RP component is supported on.

Table 6-3 MLS-RP Supported Platforms

Platform	Software Requirements
3600	12.0(2)
4500/4700 7200/7500	11.3(2)WA4(4)
Catalyst 5000 RSM/RSFC	12.0(3c)W5(8a)
Catalyst 6000 MSFC	All software trains

NOTE

It is important to note that the Catalyst 6000/6500 Supervisor 1A with PFC can act only as an MLS-SE in conjunction with the Catalyst 6000/6500 MSFC; in this configuration, the MLS-SE (the PFC) and MLS-RP (the MSFC) do not communicate over IP, instead communicating via an internal bus. However, the MSFC can act as an MLS-RP with other MLS-SEs such as a Catalyst 5000 with NetFlow Feature Card (NFFC).

Scenario Prerequisites

To successfully commence the configuration tasks required to complete this scenario, Table 6-4 describes the prerequisite configurations required on each device in the scenario topology. Any configurations not listed can be assumed as being the default configuration.

Table 6-4 Scenario 6-1 Requirements

Device	Required Configuration	
	Parameter	**Value**
Switch-A	Hostname	Switch-A
	sc0 IP Address (VLAN)	192.168.1.10/24 (VLAN 1)
	Enable/Telnet Password	cisco
	VTP Mode	Transparent
	VTP Domain Name	lanps
	VLANs (Name)	VLAN 2 (VLAN02)
	VLAN Assignments	VLAN 2: port 2/3
	ISL Trunks (DTP Mode)	2/1 (nonegotiate)
Router-A	Hostname	Router-A
	Enable/Telnet Password	cisco
	ISL Trunks	FastEthernet0/0
	ISL Trunk Subinterfaces	FastEthernet0/0.1 (VLAN 1)
	IP Address	FastEthernet0/0.2 (VLAN 2)
	(Interface)	192.168.1.1/24 (fa0/0.1) 192.168.2.1/24 (fa0/0.2)
Host-X	Operating System	Windows 2000 Professional or Windows XP
	IP Address	192.168.1.100/24
	Default Gateway	192.168.1.1
Host-Y	Operating System	Windows 2000 Professional or Windows XP
	IP Address	192.168.2.100/24
	Default Gateway	192.168.2.1
	Applications	Telnet server (e.g. Microsoft Telnet Server)

Example 6-1 and Example 6-2 shows the configuration required on Switch-A and Router-A before you can begin this scenario.

Example 6-68 Scenario 6-1 Prerequisite Configuration for Switch-A

```
Console> (enable) set system name Switch-A
System name set.
Switch-A> (enable) set password
Enter old password: _
Enter new password: *****
Reenter new password: *****
Switch-A> (enable) set enablepass
Enter old password: _
Enter new password: *****
Retype new password: *****
Switch-A> (enable) set interface sc0 192.168.1.10 255.255.255.0
Interface sc0 IP address and netmask set.
Switch-A> (enable) set vtp mode transparent
VTP domain modified
Switch-A> (enable) set vtp domain lanps
VTP domain lanps modified
Switch-A> (enable) set vlan 2 name VLAN2
Vlan 2 configuration successful
Switch-A> (enable) set trunk 2/1 nonegotiate isl
Port(s) 2/1 trunk mode set to nonegotiate.
Port(s) 2/1 trunk type set to isl.
Switch-A> (enable) set vlan 2 2/3
VLAN 2 modified.
VLAN 1 modified.
VLAN Mod/Ports
---- ----------------------
2   2/3
```

Example 6-69 Scenario 6-1 Prerequisite Configuration for Router-A

```
Router# configure terminal
Router(config)# hostname Router-A
Router-A(config)# enable secret cisco
Router-A(config)# line vty 0 4
Router-A(config-line)# password cisco
Router-A(config-line)# exit
Router-A(config)# interface FastEthernet0/0
Router-A(config-if)# no shutdown
Router-A(config-if)# exit
Router-A(config)# interface FastEthernet0/0.1
Router-A(config-if)# encapsulation isl 1
Router-A(config-if)# ip address 192.168.1.1 255.255.255.0
Router-A(config-if)# exit
Router-A(config)# interface FastEthernet0/0.2
Router-A(config-if)# encapsulation isl 2
Router-A(config-if)# ip address 192.168.2.1 255.255.255.0
```

After the prerequisite configuration is implemented, you should attach each device as indicated in Figure 6-9 and verify PING connectivity between devices in the network before proceeding.

Configuration Tasks

As you have learned, MLS consists of an MLS-SE and MLS-RP, which must each be configured separately. Configuring MLS requires the following configuration tasks:

- Configuring the MLS-RP
- Configuring the MLS-SE
- Verifying MLS operation

Configuring the MLS-RP

When configuring MLS on the MLS-RP, you must configure MLS globally on the router, which then enables you to configure global MLS parameters as well as specific interfaces for MLS. Before you can enable any MLS interface, you must configure the same VTP domain used by the MLS-SE on each MLS-enabled interface. This is because MLSP communications between MLS components cannot cross VTP domain boundaries. You must also configure an MLS management interface, which is the interface used to send and receive MLSP messages. Once you have completed all these tasks, you can then enable MLS on the required interfaces. In summary, this requires the following configuration tasks:

- Enabling MLS globally
- Configuring MLS on interfaces

Enabling MLS Globally

The first configuration task for configuring MLS on an MLS-RP is to enable MLS globally, which enables further configuration of specific MLS parameters. To enable MLS globally, the **mls rp ip** global configuration command is used, as demonstrated in Example 6-3 on Router-A.

Example 6-70 Enabling IP MLS on Router-A

```
Router-A# configure terminal
Router-A(config)# mls rp ip
```

Configuring MLS on Interfaces

After enabling MLS globally on the MLS-RP, you next configure MLS on each interface that needs to communicate with MLS-SEs. Each interface that needs to communicate with an MLS-SE requires the following configuration:

- The VTP domain of the MLS-SE switches that the MLS-RP communicates with via the interface must be configured.
- MLS must be explicitly enabled on the interface.

An MLS-RP also requires a single interface to be designated as a *management interface*, which defines the interface used for MLS-RP λν MLS-SE communications. Without a management interface, the MLS-RP does not function.

To configure MLS on an interface, the **mls** interface configuration command is used, which has the following syntaxes:

```
Switch(config-if)# mls rp vtp-domain vtp-domain-name
Switch(config-if)# mls management-interface
Switch(config-if)# mls rp ip
```

The following describes each of the command syntaxes listed above:

- **mls rp vtp-domain—**Defines the VTP domain of the MLS-SEs connected to the interface. The configured VTP domain must match the VTP domain configured on MLS-SEs; otherwise, MLS communications fail.
- **mls management-interface—**Defines the interface as the MLS management interface. Only one interface can be configured as a management interface.
- **mls rp ip—**Enables MLS operation on the interface.

WARNING

Always configure the VTP domain first, *before* configuring any other MLS commands on an interface. If you do not specify the VTP domain first, as soon as you configure another MLS command (e.g., enable MLS or configure a management interface), a null VTP domain will be assumed and configured. To change the VTP domain, all MLS interface configuration must be removed. Hence, to save time and frustration, always configure the appropriate VTP domain first.

In this scenario, Router-A has two subinterfaces configured (fastEthernet0/0.1 and fastEthernet0/0.2) over a physical trunk to Switch-A, which belong in VLAN 1 and VLAN 2 respectively. MLS must be enabled on both these interfaces, and a management interface must also be configured. Example 6-4 demonstrates enabling MLS on both interfaces and configuring fastEthernet0/0.1 as the management interface.

Example 6-71 Enabling IP MLS on MLS-RP Interfaces

```
Router-A# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router-A(config)# interface fastEthernet0/0.1
Router-A(config-if)# mls rp vtp-domain lanps
Router-A(config-if)# mls management-interface
Router-A(config-if)# mls rp ip
Router-A(config-if)# exit
Router-A(config)# interface fastEthernet0/0.2
Router-A(config-if)# mls rp vtp-domain lanps
Router-A(config-if)# mls rp ip
```

In Example 6-4, notice that the VTP domain is configured first with a value of *lanps***,** which matches the VTP domain configured on Switch-A in the "Configuration Prerequisites" section earlier in this chapter. The fastEthernet0/0.1 interface is specified as the management interface, meaning Router-A must be configured as an MLS-RP with an IP address of 192.168.1.1 on the MLS-SE.

Configuring the MLS-SE

After you have configured the MLS-RP, you can then configure the MLS-SE. Configuring the MLS-SE requires the following configuration tasks:

- Configuring VTP
- Enabling MLS
- Configuring optional MLS parameters

Configuring VTP

As indicated in the previous section on configuring the MLS-RP, the VTP domains configured on the MLS-RP and MLS-SE must match. In the configuration prerequisites section of this scenario, VTP was configured on Switch-A with a VTP domain name of *lanps*, which matches the VTP domain configured on the MLS-RP earlier, ensuring MLS communications succeed.

Enabling MLS

After ensuring VTP is configured correctly, you can begin to configure MLS. The first MLS configuration task is to enable MLS globally on the switch, which enables MLS-SE functionality. If the MLS-RP is integrated with the Catalyst switch (i.e., the RSM/RSFC on the Catalyst 5000/5500 or the MSFC on the Catalyst 6000/6500), MLS is already enabled automatically and you do not need to configure MLS, unless you wish to modify certain MLS parameters. If the MLS-RP is external (as is the case for this scenario), you must explicitly enable MLS and also define at least one MLS-RP.

To enable MLS, the **set mls enable** command is used without any other parameters. After enabling MLS, you can then define up to 16 external MLS-RPs using the **set mls include** command:

```
Console> (enable) set mls include mls-rp-address
```

The *mls-rp-address* parameter must be the IP address of the MLS management interface configured on the MLS-RP. Example 6-5 demonstrates enabling MLS on Switch-A and then defining Router-A as an MLS-RP.

Example 6-72 Enabling IP MLS and Configuring an MLS-RP on CatOS

```
Switch-A> (enable) set mls enable
IP Multilayer switching is enabled.
Switch-A> (enable) set mls include 192.168.1.1
Multilayer switching is enabled for router 192.168.1.1
```

In Example 6-5, notice that after MLS is enabled, the VLAN 1 interface IP address on Router-A is defined as an MLS-RP, which is the management interface on Router-A (see Example 6-4).

Configuring Optional MLS Parameters

At this stage, Switch-A has been configured as a fully functional MLS-SE and begins to use MLS as Router-A is operational as an MLS-RP. You can also configure other optional MLS parameters, which mainly affect the how the flow cache is operated and maintained. Some of these optional parameters include the following:

- Configuring the minimum flow mask
- Configuring MLS timers

Configuring the Minimum Flow Mask

The first optional parameter defines the *minimum* flow mask for the flow cache. By default, a flow mask of *destination* is used, which means flow entries are generated on a per-destination IP address basis. As you have already learned, three different flow masks exist—*destination*, *source-destination,* and *full flow*. You normally don't need to modify the default flow mask of *destination*, as the configuration of features that require a higher resolution masks, such as implementing ACLs on the MLS-RP, is performed automatically. If you do wish to configure the minimum flow mask to a higher resolution flow mask, you can use the **set mls flow** command:

```
Console> (enable) set mls flow {destination | destination-source | full]
```

Example 6-6 demonstrates configuring the minimum flow mask as source and destination IP address on Switch-B, which means a new flow entry is generated for each unique source and destination pair of IP addresses.

Example 6-73 Configuring the Minimum Flow Mask on CatOS

```
Switch-A> (enable) set mls flow destination-source
Configured flow mask is set to destination-source flow.
```

WARNING

The full flow mask generates a lot of entries in the MLS cache and should be used with caution. For example, in this scenario, an HTTP connection between Host-X and Host-Y is represented by a separate flow from an FTP connection between Host-X and Host-Y.

Configuring MLS Timers

The flow cache is a finite resource that only can maintain a certain amount of flow entries before the cache becomes full. If the cache becomes full, the MLS-SE can no longer write new entries for new flows, meaning any new flows that require routing cannot be Layer 3 switched. Instead they are routed normally via the router-on-stick topology of the MLS-RP. The MLS cache can accommodate 128 K (128 * 1024) entries; however, when the number of entries is above 32 K, there is a chance that the MLS-SE forwards some flows to the MLS-RP for forwarding. To avoid the flow cache from exceeding 32 K entries, the MLS-SE operates two timers, which are both used to age out idle flow entries after a configurable period of time:

- **MLS fast aging timer—**Used to age out flows that have not exceeded sending a configurable number of packets (the *packet threshold*) within the fast aging timer. By default, the fast aging timer is set to 0 (not used) and is normally configured only to reduce the size of the MLS cache when it is consistently exceeding 32K entries. You can configure the fast aging timer as 32, 64, 96, or 128 seconds; you can configure a packet threshold of 0, 1, 3, 7, 15, 31, or 63 packets. For example, if you configured a fast aging timer of 32 seconds and a packet threshold of 15, any flow that does not send more than 15 packets within 32 seconds is aged out. If the MLS cache exceeds 32 K, reduce the fast aging timer; if the MLS cache continues to exceed 32 K, you should decrease the MLS aging timer (described next).
- **MLS aging timer—**Used to age out idle flows that have not sent a single packet (if one or more packets is sent, the aging timer is reset) during the aging timer interval. The MLS aging timer is 256 seconds by default and the aging time can be configured in 8-second increments between 8 and 2032 seconds.

NOTE

You should only ever modify the MLS aging time after you have first tuned the IP MLS fast aging timer.

To configure the MLS fast aging timer and aging timer, the **set mls agingtime** command is used:

```
Console> (enable) set mls agingtime {aging-timer | fast
 fast-aging-timer packet-threshold}
```

You configure the MLS aging timer by just specifying an aging timer value, while you configure the MLS fast aging timer by specifying the **fast** keyword and then configuring the fast aging timer and packet threshold. Example 6-7 demonstrates configuring the MLS fast aging timer and aging timer on Switch-A.

Example 6-74 Configuring the MLS Aging Timer and MLS Fast Aging Timer on CatOS

```
Switch-A> (enable) set mls agingtime 480
Multilayer switching aging time set to 480
Switch-A> (enable) set mls agingtime fast 128 7
Multilayer switching fast aging time set to 128 seconds for entries
  with no more than 7
packets switched.
```

In Example 6-7, the MLS aging timer is set to 480 seconds, meaning any flow entry that is idle for 480 seconds is aged out. The MLS fast aging timer is also configured, aging out any flow entry that does not send more than 7 packets within a 128 second period.

Verifying MLS Operation

At this stage, you have configured an MLS-RP and MLS-SE, configured a minimum flow mask on the MLS-SE so that the MLS cache on the MLS-SE contains flows that represent unique source and destination IP address combinations and also configured MLS timers on the MLS-SE. To verify the MLS configuration

on the MLS-RP and MLS-SE, Host-X and Host-Y need to communicate so that MLS operation can be observed and verified.

Before testing inter-VLAN communications, it is a good idea to verify the MLS configuration on the MLS-RP and MLS-SE. On the MLS-RP (Cisco IOS-based), the **show mls rp** command can be used to verify MLS configuration. Example 6-8 demonstrates using this command on Router-A after both the MLS-RP and MLS-SE have been configured.

Example 6-75 Verifying MLS Configuration on the MLS-RP

```
Router-A# show mls rp
ip multilayer switching is globally enabled
ipx multilayer switching is globally disabled
ipx mls inbound acl override is globally disabled
mls id is 0010.7be2.aba0
mls ip address 192.168.1.1
mls ip flow mask is destination
mls ipx flow mask is unknown
number of domains configured for mls 1

vlan domain name: lanps
  current ip flow mask: destination-source
  ip current/next global purge: false/false
  ip current/next purge count: 0/0
  current ipx flow mask: destination
  ipx current/next global purge: false/false
  ipx current/next purge count: 0/0
  current sequence number: 4280145038
  current/maximum retry count: 0/10
  current domain state: no-change
  domain uptime: 00:14:43
  keepalive timer expires in 9 seconds
  retry timer not running
  change timer not running
  fcp subblock count = 2

  1 management interface(s) currently defined:
   vlan 1 on fastEthernet0/0.1

  2 mac-vlan(s) configured for multi-layer switching

  2 mac-vlan(s) enabled for ip multi-layer switching:

   mac 0010.7be2.aba0
     vlan id(s)
     1  2

  0 mac-vlan(s) enabled for ipx multi-layer switching:

  router currently aware of following 1 switch(es):
   switch id 0030.f2b8.3fff
```

Notice in Example 6-8 that IP MLS is enabled on Router-A, and that the IP address of the MLS-RP is 192.168.1.1. You can see that interface fastEthernet0/0.1 is the management interface and that two VLANs are enabled for MLS (VLAN 1 and VLAN 2, which are provided physically by fastEthernet0/0.1 and fastEthernet0/0.2, respectively). You can also see that a single VTP domain called *lanps* is configured and that Router-A is aware of Switch-A as an MLS-SE. Notice that the flow mask is indicated as being destination, as opposed to destination-source as has been configured on Switch-A. This is because Router-A has no knowledge of the flow mask on Switch-A and, therefore, has a flow mask that is independent of the switch.

You can also you the **show mls ip** command on the MLS-SE (CatOS). Example 6-9 demonstrates using this command on Switch-A after both the MLS-RP and MLS-SE have been configured.

Example 6-76 Verifying MLS Configuration on the MLS-SE

```
Switch-A> (console) show mls ip
IP Multilayer switching enabled
IP Multilayer switching aging time = 480 seconds
IP Multilayer switching fast aging time = 128 seconds, packet threshold = 7
IP Current flow mask is Destination-source flow
Configured flow mask is Destination-source flow
Active IP MLS entries = 1
Netflow Data Export disabled
Netflow Data Export port/host is not configured.
Total packets exported = 0

IP MLS-RP IP   MLS-RP ID  XTAG MLS-RP MAC-Vlans
---------------- ------------ ---- ---------------------------------
192.168.1.1   00107be2aba0  2 00-10-7b-e2-ab-a0 1-2
```

In Example 6-9, you can see that MLS is enabled and can also see the aging timers configured for MLS. Notice that the flow mask on the MLS-SE is destination-source, because this has been configured as the minimum flow mask. The last line in the output of Example 6-9 shows the IP address, MAC address, XTAG, and VLANs for which each interface of the MLS-RP is configured.

Now that the MLS configuration on both the MLS-RP and MLS-SE has been verified, it is time to actually test that inter-VLAN is routing and that MLS is being performed as it should. Before testing inter-VLAN routing, it is a good idea to clear any current flow entries from the MLS cache using the **clear mls entry ip all** command so that you can easily see when entries are generated in response to various events. Example 6-10 demonstrates clearing all current entries from the MLS cache on the MLS-SE.

Example 6-77 Clearing the MLS Cache on Switch-A

```
Switch-A> (enable) clear mls entry ip all
Multilayer switching entries cleared.
Switch-A> (enable) show mls entry ip
                 Last  Used
Destination IP Source IP    Prot DstPrt SrcPrt Destination Mac  Vlan Port
--------------- --------------- ---- ------ ------ ----------------- ---- -----
MLS-RP 192.168.1.1:
No entries
```

In Example 6-10, after all entries in the MLS cache are cleared, the **show mls entry ip** command is used to verify no entries exist in the cache.

After the MLS cache is emptied, the process of testing inter-VLAN routing can now begin. To simulate traffic that requires inter-VLAN routing, Host-X (in VLAN 1) can be used to ping Host-Y (in VLAN 2), which causes inter-VLAN routing between VLAN 1 and VLAN 2 to occur. Example 6-11 demonstrates pinging Host-Y from Host-X.

Example 6-78 Generating Inter-VLAN on Host-X

```
C:\> ping 192.168.2.100

Pinging 192.168.2.100 with 32 bytes of data:

Reply from 192.168.2.100: bytes=32 time=4ms TTL=128
Reply from 192.168.2.100: bytes=32 time<1ms TTL=128
```

```
Reply from 192.168.2.100: bytes=32 time<1ms TTL=128
Reply from 192.168.2.100: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.100:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
  Minimum = 1ms, Maximum = 4ms, Average = 2ms
```

In Example 6-11, the first packet sent by Host-X is routed through the MLS-RP, with the MLS-SE learning the information required for Layer 3 switching based upon the routed Ethernet frame that is sent from the MLS-RP to Host-Y in VLAN 2. Each subsequent packet is then Layer 3 switched in hardware on the MLS-SE, ensuring high performance inter-VLAN routing. Example 6-12 demonstrates verifying that a flow entry has indeed been generated on the MLS-SE (Switch-A) using the **show mls entry ip** command, after traffic has been generated in Example 6-11 between Host-X and Host-Y.

Example 6-79 Viewing the MLS Cache on Switch-A

```
Switch-A> (enable) show mls entry ip
                 Last  Used
Destination IP Source IP    Prot DstPrt SrcPrt Destination Mac  Vlan Port
-------------- -------------- ---- ------ ------ ----------------- ---- -----
MLS-RP 192.168.2.2:
 No entries
MLS-RP 192.168.1.1:
192.168.1.100  192.168.2.100  -  -   -   00-10-a4-e0-1e-d3 1  2/2
192.168.2.100  192.168.1.100  -  -   -   00-06-53-fe-84-20 2  2/3
```

Notice in Example 6-12 that two new entries are in the MLS cache for the MLS-RP at 192.168.1.1 (Router-A). The first entry indicates that any IP packet with a source IP address of 192.168.2.100 (Host-Y) and a destination IP address of 192.168.1.100 (Host-X) should be L3 switched by rewriting the destination MAC address of the frame to 00-10-a4-e0-1e-d3 (the MAC address of Host-X) and then switching the frame out port 2/2 (which is attached to Host-X). Similarly, the second entry describes the information required to L3 switch traffic from Host-X (192.168.1.100) to Host-Y (192.168.2.100). Example 6-12 demonstrates that a bidirectional communications session between two devices generates two flow entries—one for each direction packets are sent.

You can see that the flow entries generated on the MLS-SE in Example 6-12 have a flow mask of destination-source, which means only a single pair of flow entries are generated for any communications between Host-X and Host-Y. If on Host-X you attempt to establish another form of connectivity to Host-Y (e.g., access a file share or establish a Telnet connection), no new flow entries are generated.

Verifying MLSP Operation

MLSP is used to communicate information between the MLS-RP and MLS-SE. Events such as routing topology changes and access control list configuration changes are communicated to the MLS-SE by the MLS-RP via MLSP messages. In the event of a routing topology change, the MLS-SE flushes the MLS cache to ensure that packets are not L3 switched in error. In the event of an access control list (ACL) being applied to an interface on the MLS-RP, the MLS-SE adjusts the flow mask to that specified by the MLS-RP to ensure traffic cannot bypass the ACL.

Example 6-13 demonstrates configuring an extended ACL on Router-A and then applying it to interface fastEthernet0/0.1, which is attached to VLAN 1 on Switch-B. After this configuration is implemented, Router-A (as the MLS-RP) should immediately notify Switch-A (the MLS-SE) that the flow mask needs to be modified.

Example 6-80 Configuring an Extended Access Control List on Switch-A

```
Router-A# configure terminal
Router-A(config)# access-list 100 permit tcp any any eq telnet
Router-A(config)# access-list 100 deny ip any any log
Router-A(config)# interface fastEthernet0/0.1
Router-A(config-if)# ip access-group 100 in
```

In Example 6-13, the ACL configured permits only Telnet traffic, denying all other traffic. After applying the ACL to interface fastEthernet0/0.1, Router-A communicates via MLSP to the MLS-SE that the flow mask needs to be modified to a full flow mask to ensure the MLS-SE does not permit other types of traffic. Example 6-14 demonstrates checking the MLS configuration on Switch-A after the ACL is applied on Router-A in Example 6-13.

Example 6-81 Checking the MLS Cache on Switch-A

```
Switch-A> (console) show mls ip
Total packets switched = 1748724
Total Active MLS entries = 0
IP Multilayer switching enabled
IP Multilayer switching aging time = 480 seconds
IP Multilayer switching fast aging time = 128 seconds, packet threshold = 7
IP Current flow mask is Full flow
Configured flow mask is Destination-source flow
Active IP MLS entries = 0
```

In Example 6-14, notice that the configured flow mask is destination-source; however, the current flow mask is full, which is due to the ACL configuration on Router-A causing Router-A to indicate to Switch-A that the flow mask needs to be changed to a full flow mask.

At this point, Host-X is able to establish only Telnet connections to Host-Y based upon the ACL configured in Example 6-13 (ping requests should fail). Because a full flow mask is now configured on Switch-A, a new flow entry should be generated for each connection routed through the Layer 3 switching engine of Switch-A. Figure 6-12 demonstrates establishing two Telnet connections to Host-Y from Host-X, and then Example 6-15 demonstrates checking the MLS cache after the connections have been established.

Figure 6-12 Establishing Two Telnet Connections From Host-X to Host-Y

Example 6-82 Viewing the MLS Cache on Switch-A

```
             Switch-A> (enable) show mls entry ip
Destination IP Source IP    Prot DstPrt SrcPrt Destination Mac  Vlan Port
--------------- --------------- ---- ------ ------ ----------------- ---- -----
MLS-RP 192.168.2.2:
 No entries
MLS-RP 192.168.1.1:
192.168.2.100  192.168.1.100  TCP Telnet 3779  00-06-53-fe-84-20 2  2/3
192.168.2.100  192.168.1.100  TCP Telnet 3780  00-06-53-fe-84-20 2  2/3
192.168.1.100  192.168.2.100  TCP 3780  Telnet 00-10-a4-e0-1e-d3 1  2/2
192.168.1.100  192.168.2.100  TCP 3779  Telnet 00-10-a4-e0-1e-d3 1  2/2
```

Notice in Example 6-15 that four new flows are cached in the MLS cache, one for each direction of traffic for each connection. Because the source TCP port (on Host-X) of each connection is different, separate pairs of flow entries in the MLS cache are present due to the flow mask being now configured as full (see Example 6-14). With the ACL configured, you should never see any completed flow entries for traffic other than Telnet traffic. Because the access list on Router-A drops all non-Telnet traffic, Switch-A never sees any traffic associated with non-Telnet packets return from Router-A and thus never completes a flow entry for the non-Telnet traffic in the MLS cache.

Scenario 6-2: Configuring CEF-based Layer 3 Switching on the Catalyst 6000/6500 Operating in Hybrid Mode

The next-generation of Cisco Catalyst Layer 3 switches are all based upon Cisco Express Forwarding (CEF). CEF is also the next-generation route caching mechanism for interrupt context switching on Cisco routers so understanding how CEF works and how to configure it is important for network engineers. CEF offers significant improvements over MLS, the most notable being that the first packet in a flow does not need to process-switched by the control plane routing component, as is the case with MLS. With CEF, all packets (even the first) associated with a flow are Layer 3 switched in hardware. This is an important consideration in environments where many new flows are being established continuously (e.g., an Internet service provider environment), because large amounts of new flows in an MLS configuration reduces performance.

In this scenario you learn how to configure Layer 3 switching using CEF on a *hybrid mode* Catalyst 6000/6500 switch, which refers to a configuration where the Supervisor engine runs CatOS and the MSFC runs a separate Cisco IOS operating system. In later scenarios, you learn how to upgrade from a hybrid mode system to a native mode system and then configure Layer 3 switching using native mode.

Figure 6-13 shows topology used for this scenario.

In Figure 6-13, Switch-A is a Catalyst 6509 switch a Supervisor 2 engine installed, which includes a PFC-2 and MSFC-2. Switch-A is running hybrid mode, with CatOS running on the Supervisor 2 engine and Cisco IOS running on the MSFC-2. The goal of this scenario is simply to enable inter-VLAN routing between Host-X and Host-Y, using CEF-based Layer 3 switching.

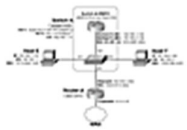


Figure 6-13 Topology for Scenario 6-2

Introduction to Cisco Express Forwarding

Cisco Express Forwarding (CEF) allows the appropriate information required for the data plane operations of Layer 3 routing (e.g., MAC address rewrites on an Ethernet network and determining the egress port out which a routed frame should be sent) to be stored in a compact data structure optimized for fast lookup.

NOTE

CEF is not only used by Cisco Catalyst Layer 3 switches; it is also used by Cisco routers (in fact, CEF was originally developed on high-end Cisco routers). CEF represents the recommended "route caching" mechanism that should be configured on all Cisco Layer 3 devices if possible.

You have learnt that MLS uses a *flow-based* caching mechanism, where packets must first be process-switched by the MLS-RP to generate flow entries in the cache. In environments where thousands of new flows are being created per second, this can cause the MLS-RP to become a bottleneck. CEF was developed to eliminate the performance penalty of the first-packet process-switched lookup, allowing the route cache used by the hardware-based L3 routing engine to contain all the necessary information to L3 switch in hardware before any packets associated with a flow are received. To achieve this, CEF creates two data structures in the route cache:

- **Forwarding Information Base (FIB)—**The FIB is generated directly from the route table and contains the next-hop IP address information for each destination (IP route) in the network.
- **Adjacency table—**The adjacency table defines each next-hop IP address in terms of MAC address and egress interface. MAC address information is collected via the ARP cache.

Figure 6-14 illustrates how the route table, FIB, adjacency table and ARP cache work together.

Figure 6-14 CEF Components

In Figure 6-14, the route table and ARP cache are both control plane entities, meaning that both are generated and maintained via the control plane processor. From these tables, the FIB and adjacency tables are built, which are data structures optimized for fast lookup by the data plane processor In Figure 6-14, the Layer 3 switching engine uses the FIB and adjacency table to determine the MAC address of the next hop device for a packet, providing the Layer 3 switching engine the required information for MAC address to rewritten in hardware rather than software. Notice how the FIB is built in the route table; the destination address, subnet mask (or prefix), and next hop gateway are extracted from the route table (the remaining information in the route table is not relevant to the Layer 3 switching process). The adjacency table then builds the MAC address details that are used to rewrite the destination MAC address of each Layer 3 switched frame, using the ARP cache (populated via the control plane using ARP requests).

NOTE

The MAC address of the router must also be known to the Layer 3 switching engine, which is used to rewrite the source MAC address of the Layer 3 switched frame. This MAC address is always the same value and hence does not need to be included in the FIB/adjacency tables.

You might notice that all of the information contained in the FIB and adjacency table is the same information contained within the route and ARP tables. The FIB and adjacency exist purely for organizing the specific information required for Layer 3 switching into a structure that is optimized for fast lookups by the data plane.

Distributed and Accelerated CEF

By default, all CEF-based Cisco Catalyst switches use a central Layer 3 switching engine to provide CEF-based Layer 3 switching, where a single processor makes all Layer 3 switching decisions for traffic received on all ports in the switch. Even though the Layer 3 switching engines used in Cisco Catalyst switches provide high performance, in some networks, having a single Layer 3 switching engine to all the Layer 3 switching does not provide sufficient performance. *Distributed CEF (dCEF)* and *Accelerated CEF (aCEF)* are technologies that implement multiple Layer 3 switching engines so that simultaneous Layer 3 switching operations can occur in parallel, boosting overall system performance.

dCEF can be used in Cisco Catalyst 6500 switches and refers to the use of multiple CEF tables distributed across multiple line cards installed in the chassis. Each line card that supports dCEF has its own dedicated hardware-based L3 routing engine and CEF route cache, allowing for multiple L3 data plane operations to be performed simultaneously within a single chassis-based system. The main route processor of the switch is responsible for generating a central master FIB and adjacency table and distributing these tables out to each dCEF line card. Figure 6-15 demonstrates the dCEF architecture.

Figure 6-15 L3 Switching Using dCEF

aCEF is a new feature supported in conjunction with the new Supervisor 720 engine which works in a similar fashion to MLS. In MLS, line cards send the initial packet of a flow to the Supervisor engine, where the packet is switched in hardware using the master CEF table. The forwarding decision made is then stored in a local scaled-down CEF table on the line card where the flow enters the switch, with the local line card making any subsequent forwarding decisions for packets associated with a flow in the local CEF table.

The Cisco CEF Implementation

The use of CEF-based L3 switching is supported on the following Cisco Catalyst switches:

- Catalyst 3550/3750 series with L3 enhanced image
- Catalyst 4000/4500 series with Supervisor 3/4
- Catalyst 6000/6500 series with Supervisor 2, PFC-2, and MSFC-2
- Catalyst 6000/6500 series with Supervisor 720 (includes PFC-3 and MSFC-3)

NOTE

CEF is also supported on all Cisco routers from IOS 12.0 upwards. It is important to understand that the CEF implementation on low-end routers (i.e., Cisco 3700 series and lower) still uses the main processor for data plane operations, whereas L3 switches use a dedicated hardware chip (ASIC) for data plane operations (e.g., PFC-2) instead of the main processor. Using CEF still provides a performance advantage over older route caching technologies, such as fast switching. On higher end routers (e.g., 7500 series, 10000 series, and 12000 series), data plane operations are performed in specialized hardware-based ASICs.

Scenario Prerequisites

To successfully commence the configuration tasks required to complete this scenario, Table 6-5 describes the prerequisite configurations required on each device in the scenario topology. Any configurations not listed can be assumed as being the default configuration.

Table 6-5 Scenario 6-2 Requirements

Device	Required Configuration	
	Parameter	**Value**
Switch-A	Hostname	Switch-A
	sc0 IP Address (VLAN)	192.168.1.10/24 (VLAN 1)
	Enable/Telnet Password	cisco
	VTP Mode	Transparent
	VLANs (Name)	VLAN 2 (VLAN02)
	VLAN Assignments	VLAN 2: port 2/2
Router-A	Hostname	Router-A
	Enable/Telnet Password	cisco
	IP Address (Interface)	192.168.1.2/24 (Ethernet0/0)
Host-X	Operating System	Windows 2000 Professional or Windows XP
	IP Address	192.168.1.100/24
	Default Gateway	192.168.1.1
Host-Y	Operating System	Windows 2000 Professional or Windows XP
	IP Address	192.168.2.100/24
	Default Gateway	192.168.2.1

Example 6-16 and Example 6-17 shows the configuration required on Switch-A and Router-A before you can begin this scenario.

Example 6-83 Scenario 6-2 Prerequisite Configuration for Switch-A

```
Console> (enable) set system name Switch-A
System name set.
Switch-A> (enable) set password
Enter old password: ø
Enter new password: *****
Reenter new password: *****
Switch-A> (enable) set enablepass
Enter old password: ø
Enter new password: *****
Retype new password: *****
Switch-A> (enable) set interface sc0 192.168.1.10 255.255.255.0
Interface sc0 IP address and netmask set.
Switch-A> (enable) set vtp mode transparent
VTP domain modified
Switch-A> (enable) set vlan 2 name VLAN2
```

```
Vlan 2 configuration successful
Switch-A> (enable) set vlan 2 2/2
VLAN 2 modified.
VLAN 1 modified.
VLAN Mod/Ports
---- ----------------------
2   2/2
```

Example 6-84 Scenario 6-2 Prerequisite Configuration for Router-A

```
Router# configure terminal
Router(config)# hostname Router-A
Router-A(config)# enable secret cisco
Router-A(config)# line vty 0 4
Router-A(config-line)# password cisco
Router-A(config-line)# exit
Router-A(config)# interface Ethernet0/0
Router-A(config-if)# no shutdown
Router-A(config-if)# ip address 192.168.1.2 255.255.255.0
```

After the prerequisite configuration is implemented, you should attach each device as indicated in Figure 6-13 and verify PING connectivity between devices in the network (where possible) before proceeding.

Configuration Tasks

Configuring CEF-based L3 switching on the Catalyst 6000/6500 operating in hybrid mode is very simple. On the Supervisor 2 engine (running CatOS), all that is required is for a PFC-2 and MSFC-2 daughtercard to be installed. No CatOS software configuration is required; however, you can tune and monitor the PFC-2 from CatOS. All configuration is controlled via the Cisco IOS running on the MSFC-2, where the IP route table is generated from the various routing protocols configured on the MSFC which then automatically populates the CEF table on the PFC-2. In summary, the following is required to configure CEF-based L3 switching:

- Configuring the MSFC
- Configuring the PFC (optional)
- Verifying CEF operation

Configuring the MSFC

To configure the MSFC on a hybrid mode Catalyst 6000/6500, the following tasks are required:

- Establishing management connectivity to the MSFC
- Configuring the MSFC-2

Establishing Management Connectivity to the MSFC-2

On a hybrid mode Catalyst 6000/6500 switch, configuration of the MSFC requires a management connection to be established to the MSFC operating system (Cisco IOS), which is separate from the CatOS running on the Supervisor engine. On the Catalyst 6000/6500, the MSFC is accessible via module 15, which is an internal designation for the internal interface between the MSFC daughtercard and Supervisor engine. You can access the MSFC management interface using the **session 15** command (where 15 represents the slot number of the MSFC), or alternatively, you can also use the **switch console** command. Example 6-18 demonstrates gaining access to the MSFC from CatOS.

Example 6-85 Gaining Management Access to the MSFC

```
Switch-A> (enable) switch console
Trying Router-15...
Connected to Router-15.
Type ^C^C^C to switch back...

Router>
```

TIP

The **session** command can be used only if the Supervisor detects the MSFC. If the MSFC has problems booting (for example, the IOS image is corrupted) and boots to ROMMON mode, the Supervisor does not detect the MSFC, and you cannot use the **session** command to connect to the MSFC console. In this situation, use the **switch console** command, because this accesses a console connection that is permanently wired to an internal console port on the MSFC.

Configuring the MSFC-2

After establishing initial management connectivity to the MSFC-2, you can next configure the MSFC-2 for routing. In Example 6-18, the MSFC should have a blank configuration because it has not yet been configured and, therefore, requires not only configuration relevant to Layer 3 switching but also any relevant base configuration. Example 6-19 shows the base configuration required on the MSFC-2.

Example 6-86 Base Configuration for MSFC-2 on Switch-A

```
Router> enable
Router# configure terminal
Router(config)# hostname Switch-A-MSFC
Switch-A-MSFC(config)# enable secret cisco
Switch-A-MSFC(config)# line vty 0 4
Switch-A-MSFC(config-line)# password cisco
```

In Example 6-19, the MSFC host name, Telnet password, and enable password is configured.

After the base configuration is complete, Layer 3 interfaces need to be created so that IP routing can take place. In this scenario, two Layer 3 switched virtual interfaces (SVIs) are required, which each attach to VLAN 1 and VLAN 2, respectively. Example 6-20 demonstrates configuring the required SVIs on the MSFC.

Example 6-87 Configuring SVIs on the MSFC

```
Switch-A-MSFC# configure terminal
Switch-A-MSFC(config)# interface VLAN 1
Switch-A-MSFC(config-if)# no shutdown
Switch-A-MSFC(config-if)# ip address 192.168.1.1 255.255.255.0
Switch-A-MSFC(config-if)# exit
Switch-A-MSFC(config)# interface VLAN 2
Switch-A-MSFC(config-if)# ip address 192.168.2.1 255.255.255.0
```

At this stage, you should be able to ping the sc0 interface (192.168.1.10) on the Supervisor engine of Switch-A because the VLAN 1 interface in Example 6-20 has been configured with an IP address. You should also be able to ping Router-A (192.168.2.2), Host-X, and Host-Y from Switch-A assuming these devices are connected and configured as per the earlier configuration prerequisites section.

Once IP connectivity has been verified, you can then configuring Layer 3 routing on the MSFC. In this scenario, the MSFC is part of an OSPF routing domain and must be configured to exchange routes with Router-A. Example 6-21 demonstrates configuring OSPF on the MSFC.

Example 6-88 Configuring IP Routing on the MSFC

```
Switch-A-MSFC# configure terminal
Switch-A-MSFC(config)# router ospf 1
Switch-A-MSFC(config-router)# network 192.168.1.0 0.0.0.255 area 0
Switch-A-MSFC(config-router)# network 192.168.2.0 0.0.0.255 area 0
Switch-A-MSFC(config-router)# end
Switch-A-MSFC# show ip ospf neighbors

Neighbor ID   Pri  State      Dead Time  Address     Interface
192.168.1.2    1  FULL/BDR    00:00:38  192.168.1.2   Vlan1
Switch-A-MSFC# show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
    D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
    N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
    E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
    i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
    * - candidate default, U - per-user static route, o - ODR
    P - periodic downloaded static route

Gateway of last resort is not set

O IA 10.0.0.0/8 [110/2] via 192.168.1.2, 00:00:32, Vlan1
C  192.168.1.0/24 is directly connected, Vlan1
C  192.168.2.0/24 is directly connected, Vlan2
```

In Example 6-21, both VLAN interfaces are configured as part of OSPF area 0. After the configuration, the **show ip ospf neighbors** command is executed, which verifies an OSPF adjacency has formed with Router-A. The **show ip route** command verifies that Switch-A-MSFC has learned the 10.0.0.0/8 network from Router-A.

Configuring the PFC (Optional)

No configuration is required of the PFC using the Supervisor engine to enabled CEF operation. CEF-based Layer 3 switching is solely enabled by configuring the MSFC.

One area where you can configure the Supervisor engine for CEF-based Layer 3 switching is in configuring *NetFlow statistics*. NetFlow is a protocol that is used in many service provider and enterprise networks which provides statistics to a NetFlow data collector about each flow or connection that passes through a routing device. This information is often for billing purposes in a service provider, making NetFlow a crucial feature of the network. NetFlow information can also be used for traffic analysis, which is common in enterprise networks. Traditionally, routers have been used for NetFlow; however, with the advent of Layer 3 switching, NetFlow has also moved to the switch. On a Layer 3 switch, the flow-based architecture of MLS (discussed in Scenario 6-1) fits well with NetFlow because the information about flow entries stored in cache can be directly exported to NetFlow. In a CEF-based Layer 3 switching architecture, however, the route caching mechanism is not flow-based, instead being based on routing topology and Layer 2 adjacency information rather than flows.

Cisco Catalyst 6000/6500 switches that perform CEF-based Layer 3 switching also collect flow information, purely for the purposes of supporting NetFlow statistics collection. On the Supervisor engine, you configure NetFlow statistics collection much like you configure MLS. The various commands that affect how information is stored in the MLS flow cache are the same commands used to determine how information is

stored for NetFlow statistics on a CEF-based switch (all NetFlow information is stored in the *NetFlow table*). These commands include the following:

- **set mls agingtime—**Used to control how long an entry remains in the NetFlow table.
- **set mls flow—**Used to control the flow mask that defines the granularity of the NetFlow table.
- **set mls exclude protocol—**Used to exclude TCP and/or UDP flows from the NetFlow table.

NOTE

You can use the **show mls statistics** command to view information about the various flows stored in the Netflow table.

Verifying CEF Operation

At this stage, the configuration of the network is complete. Host-X and Host-Y should be able to communicate with each other and also should be able to ping the loopback 0 interface on Router-A (10.0.0.1). This verifies that inter-VLAN routing is working; however, just because inter-VLAN routing is working doesn't necessarily mean that Layer 3 switching using CEF is working. Packets possibly could be routed in software via the MSFC, which of course in a production environment on a busy network degrades performance. If you are configuring CEF for Layer 3 switching, you must understand how to verify not only inter-VLAN routing, but also how to verify CEF operation.

To verify CEF operation on a hybrid mode Catalyst 6000/6500, you use various **show mls** commands, which display information relating to CEF operation. The **show mls cef mac** command can be used to view the MAC address that is used by the MSFC, which allows the PFC on the Supervisor engine to identify a candidate frame for Layer 3 switching. Example 6-22 demonstrates the use of this command.

Example 6-89 Displaying the MSFC MAC Address on CatOS

```
Switch-A> (enable) show mls cef mac
Module 15: Physical MAC-Address 00-04-dd-41-89-0a
Module 15 is the designated MSFC for installing CEF entries
```

In Example 6-22, you can see that Switch-A knows that the MAC address of the MSFC is 00-04-dd-41-89-0a, meaning any frames received with a destination MAC address of this address require Layer 3 switching.

As discussed in the introduction of this chapter, CEF uses two tables to represent control plane routing operation in a format that can be quickly looked up by the L3 engine on the PFC-2. The first table is the Forwarding Information Base (FIB). To view the FIB, you can use the **show mls entry cef ip** command, as demonstrated on Switch-A in Example 6-23.

Example 6-90 Displaying the FIB on CatOS

```
Switch-A> (enable) show mls entry cef ip
Mod FIB-Type Destination-IP Destination-Mask NextHop-IP   Weight
--- --------- --------------- ---------------- --------------- ------
 15 receive  255.255.255.255 255.255.255.255
 15 receive  127.0.0.12   255.255.255.255
 15 receive  127.0.0.0    255.255.255.255
 15 receive  127.255.255.255 255.255.255.255
 15 resolved 127.0.0.11   255.255.255.255 127.0.0.11      1
 15 receive  192.168.2.1   255.255.255.255
 15 receive  192.168.2.0   255.255.255.255
 15 receive  192.168.2.255  255.255.255.255
 15 resolved 192.168.2.100  255.255.255.255 192.168.2.100    1
 15 receive  192.168.1.1   255.255.255.255
```

```
15 receive  192.168.1.0   255.255.255.255
15 receive  192.168.1.255  255.255.255.255
15 resolved 192.168.1.100  255.255.255.255 192.168.1.100    1
15 receive  224.0.0.0    255.255.255.0
15 connected 192.168.2.0   255.255.255.0
15 connected 192.168.1.0   255.255.255.0
15 resolved 10.0.0.0    255.0.0.0    192.168.1.2     1
15 connected 127.0.0.0    255.0.0.0
15 drop   224.0.0.0    240.0.0.0
15 wildcard 0.0.0.0     0.0.0.0
```

The FIB table is generated by the MSFC from the MSFC route table and written to the CEF FIB table in the route cache on the PFC-2. In Example 6-23, the *Mod* column indicates the module number of the MSFC (module 15) that generated each FIB entry. The *FIB-Type* column describes the type of FIB entry. The following describes each of the FIB types shown in Example 6-23:

- **resolved—**Indicates an entry that represents the route to a host on the local subnet or remote destination subnet derived from the routing table on the MSFC. Every resolved FIB entry includes a next hop IP address.
- **receive—**Indicates that any packet that matches the indicated destination IP address or IP subnet should be sent to the MSFC for processing. For example, these entries are used for any management traffic to a local IP address on the MSFC.
- **connected—**Indicates an entry that represents a locally connected IP subnet.
- **drop—**Indicates that any packet that matches the indicated destination IP address or IP subnet should be dropped. In Example 6-23, because multicast routing is not enabled on the MSFC, all multicast traffic (indicated by a destination IP address in the range of 224.0.0.0 ν 239.255.255.255) is dropped.
- **wildcard—**Indicates the entry that matches any packets that do not match other FIB entries. The actions for this entry is to drop the traffic, as it is deemed unroutable.

The important entries in Example 6-23 are shaded. For example, the last shaded entry specifies a destination prefix of 10.0.0.0/8 and indicates that the next hop for this destination is 192.168.1.2 (Router-A).

The next table related to CEF is the adjacency table, which can be displayed by executing the **show mls entry adjacency** command, as shown in Example 6-24.

Example 6-91 Displaying the CEF Adjacency Table on CatOS

```
Switch-A> (enable) show mls entry cef adjacency
Mod:         15
Destination-IP:   192.168.2.100   Destination-Mask:  255.255.255.255
FIB-Type:      resolved

AdjType NextHop-IP   NextHop-Mac    Vlan Encp Tx-Packets  Tx-Octets
-------- --------------- ----------------- ---- ---- ------------ -------------
connect 192.168.2.100  00-06-53-fe-84-20  2 ARPA      4      256
*******************************************************************************
Mod:         15
Destination-IP:   127.0.0.11     Destination-Mask:  255.255.255.255
FIB-Type:      resolved

AdjType NextHop-IP   NextHop-Mac    Vlan Encp Tx-Packets  Tx-Octets
-------- --------------- ----------------- ---- ---- ------------ -------------
connect 127.0.0.11   00-00-11-00-00-00  0 ARPA      0       0
*******************************************************************************
Mod:         15
Destination-IP:   192.168.1.100   Destination-Mask:  255.255.255.255
FIB-Type:      resolved

AdjType NextHop-IP   NextHop-Mac    Vlan Encp Tx-Packets  Tx-Octets
-------- --------------- ----------------- ---- ---- ------------ -------------
connect 192.168.1.100  00-10-a4-e0-1e-d3  1 ARPA      4      256
*******************************************************************************
Mod:         15
Destination-IP:   10.0.0.0     Destination-Mask:  255.255.255.0
FIB-Type:      resolved

AdjType NextHop-IP   NextHop-Mac    Vlan Encp Tx-Packets  Tx-Octets
-------- --------------- ----------------- ---- ---- ------------ -------------
connect 192.168.1.2   00-d0-05-15-64-0a  2 ARPA      5      420
```

In Example 6-24, notice that asterisked lines are included to show each of the separate adjacency entries. The output shows each of the next hop devices listed in the FIB table (see Example 6-23) and indicates the next hop interface and the number of IP packets and bytes transmitted. For example, the last entry in Example 6-24 is for the FIB prefix 10.0.0.0/8. In the adjacency information, you can see that the next hop address is 192.168.1.2 (Router-A). This information is generated from the routing table on the MSFC. The **NextHop-Mac** address column represents the destination MAC address of the next hop router, which is required as the address used to rewrite the destination MAC address for the relevant prefix. The MAC address information is built from the ARP cache. Finally, the **Tx-Packets** and **Tx-Octets** columns indicate how many packets and bytes have been L3 switched that match the adjacency entry.

TIP

A much tidier method of displaying the CEF adjacencies is to use the **show mls entry cef long** command. This command is not demonstrated here as the output is displayed in a table format that is very wide.

Scenario 6-3: Upgrading from Hybrid Mode to Native Mode on the Catalyst 6000/6500

At the beginning of this chapter, you learned that there are two different modes in which you can operate a Cisco Catalyst 6000/6500 Layer 3 switch. The first mode is hybrid mode, where a separate and different operating system (CatOS and Cisco IOS) runs on the Supervisor engine and MSFC respectively; in the previous scenario, you learned how to configure a Catalyst 6000/6500 switch running in hybrid mode. The second mode is native mode, where a single operating system (Cisco IOS) controls and configures both the Supervisor engine and MSFC and is considered the way of the future for the Cisco Catalyst 6000/6500 switch. By default, all Cisco Catalyst 6000/6500 switches that include an MSFC ship in hybrid mode, so if you wish to run native mode, you must upgrade your switch to native mode yourself.

In this scenario, you learn how to upgrade a hybrid mode Catalyst 6000/6500 switch to native mode and also learn about some of the restrictions of running native mode. This scenario follows on from the previous scenario with the goal of ensuring the same level of functionality as the previous scenario; the only difference being that Switch-A is to operate in native mode as opposed to hybrid mode.

Understanding Native Mode

Before examining native mode, it is important to understand that the Supervisor module has its own processor, which is referred to as the *switch processor*. The switch processor is responsible for managing the Layer 2 components of the switch, as well as the PFC card that provides Layer 3/4 data plane operations. The MSFC also has its own processor, which is referred to as the *route processor*; this is responsible for managing Layer 3 control plane operations.

When a Cisco Catalyst 6000/6500 is installed with a MSFC, the switch can be described as operating in two modes:

- **Hybrid mode—**In hybrid mode (or Hybrid mode), separate operating systems manage the Supervisor module and the MSFC. The Supervisor module operating system (CatOS) has control over the Layer 2 switching functions and hardware-based L3 switching (data plane) functions, while the MSFC operating system (Cisco IOS) has control over the Layer 3 routing control plane functions. Hybrid mode is the default mode of operation, and the Supervisor and MSFC require separate management.
- **Native mode—**In native mode (or Native mode), a single operating system is used to manage both the Supervisor module and the MSFC. The operating system, which is Cisco IOS, has control over all Layer 2 and Layer 3 functions of the switch. Native mode represents the future direction of the Cisco Catalyst 6000/6500; however, it is important to note that at this time, native mode requires an MSFC to be installed alongside the Supervisor, and some line cards and features are not currently supported in native mode.

Hybrid mode represents the majority of Catalyst 6000/6500 installations today; this will remain so for the foreseeable future because there are some specific requirements and restrictions when using native mode.

Native Mode Requirements

To upgrade to native mode (if your Catalyst 6000/6500 switch has a Supervisor 1 module installed) it must meet the following prerequisites:

- Policy feature card (PFC-1) installed.
- Multilayer switching feature card (MSFC-1 or MSFC-2) installed.
- Supervisor 1 must have at least 16 MB Flash (default) and 64 MB DRAM (default) installed.

- MSFC-1 or MSFC-2 must have at least 16 MB Flash (default) and 128 MB DRAM (default for MSFC-2) installed. For larger and more complex networks, 256 MB or 512 MB of DRAM is recommended.
- A 32 MB PCMCIA flash card is recommended at least temporarily for rollback purposes, as you must format the internal flash file system to a Cisco IOS format.

If your Catalyst 6000/6500 switch has a Supervisor 2 module installed, it must meet the following prerequisites:

- Policy Feature Card (PFC-2) that is integrated into the Supervisor 2 module.
- Multilayer Switching Feature Card (MSFC-2) installed (the Supervisor 2 does not support the MSFC-1).
- Supervisor 2 must have at least 32 MB Flash and 128 MB DRAM (default) installed.
- MSFC-2 must have at least 16 MB Flash (default) and 256 MB DRAM (default) installed.
- A 32 MB PCMCIA Flash card is recommended at least temporarily for rollback purposes because you must format the internal Flash file system to a Cisco IOS format.

NOTE

If your Catalyst 6000/6500 switch has a Supervisor 720 engine installed, this engine in its default configuration meets the requirements for upgrading to native mode.

The main requirement for native mode (and also the main limiting factor for choosing to upgrade to native mode) is the MSFC. The MSFC certainly isn't cheap and is obviously not required for many LAN topologies where the switch is required to L2 switch packets and classify packets only at L3/L4 (the PFC is only required to classify packets at L3/L4).

Native Mode Limitations

It is important to understand that there are some restrictions on both the hardware and software features you can use with native mode. At the time of writing, hybrid mode supports more features than native mode, although at some stage in the not too distant future, Cisco has indicated native mode will become equal with hybrid mode in terms of features. From there native mode will be the primary development platform for the Catalyst 6000/6500.

In terms of hardware limitations, the following modules are not supported in the latest Cisco IOS release at the time of writing (12.1(14)E):

- Voice modules (WS-X6624-FXS, WS-X6608-T1, WS-X6608-E1)
- ATM LANE modules (WS-X6101-OC12-MMF, WS-X6101-OC12-SMF)
- Multilayer Switch Module (WS-X6302-MSM)
- 2-port OC-12/STM-4 ATM OSM, MM (WS-X6101-OC12-MMF)
- 2-port OC-12/STM-4 ATM OSM, SM-IR (WS-X6101-OC12-SMF)

If you have these modules installed in a native mode system, the modules remain powered down and do not interfere with the operation of the switch.

In terms of software limitations, quite a number of features are still not supported in native mode, although this list will grow smaller with each new release of Cisco IOS. Notable features that are not supported include:

- High availability
- Dynamic VLANs using VLAN Membership Policy Server (VMPS)

- Multi-Instance Spanning Tree Protocol (MISTP)
- MAC address filtering
- Layer 2 traceroute

Probably the most important feature not available is high availability. A feature called Route Processor Redundancy Plus (RPR+) is supported; however, the failover time is in the order of 30 to 60 seconds. For many Catalyst 6000/6500 installations with redundant Supervisor configurations, this failover time is unacceptable and is definitely a limiting factor in the migration to native mode. Hybrid mode supports the high availability feature, which synchronizes the various state tables between each Supervisor (e.g., spanning tree, bridge table) and allows failover within three seconds.

NOTE

Other ways of implementing faster convergence for high availability networks exist, such as using Hot Standby Routing Protocol (HSRP), which can provide sub-second failover when operating redundant Layer 3 chassis. If you want a redundancy in a single chassis, however, native mode does not support the fast failover high availability brings to hybrid mode installations.

The Hybrid Mode versus Native Mode Boot Process

It is important to understand the differences in the boot process of both hybrid mode and native mode so that you can understand the file system and image requirements of upgrading to native mode. This information also helps you to revert back to hybrid mode if required.

The Hybrid Mode Boot Process

With hybrid mode, the following describes the files that the Supervisor and MSFC modules boot from:

- **Supervisor Image—**The Supervisor CatOS image is normally installed in the *bootflash:* device (the onboard flash device) of the Supervisor and always begins with a prefix of cat6000-sup—e.g., cat6000-sup2k8.7-2-2.bin.
- **MSFC Boot Loader—**The MSFC uses two files. The first is the boot loader file, which is required to initially boot the MSFC. This file is normally stored in the *bootflash:* device of the MSFC (not the Supervisor), and begins with a prefix of c6msfc-boot (for MSFC-1) or c6MSFC-2-boot (for MSFC-2)—e.g., c6MSFC-2-boot-mz.121-8a.E4**.**
- **MSFC Image—**The MSFC Cisco IOS operating system image is normally stored in the *bootflash:* device of the MSFC and begins with a prefix of c6msfc (for MSFC-1) or c6MSFC-2 (for MSFC-2)—e.g., c6MSFC-2-jsv-mz.121-8a.E4.

When a hybrid mode switch first boots, the switch processor on the Supervisor module reads a boot environment variable, which specifies the full path to the operating system image that should be booted. This image is a CatOS-based image, which boots the Supervisor module. The Supervisor module initializes the various modules installed, runs systems diagnostics, and loads the CatOS operating system into Supervisor memory. If an MSFC is installed, once it has been initialized by the switch processor, the MSFC boot loader image is loaded, after which the main MSFC image is loaded by the route processor on the MSFC. This image is a Cisco IOS-based image which initializes the various hardware components on the MSFC, runs system diagnostics, and loads the Cisco IOS operating system into MSFC memory. Once the Supervisor and MSFC boot processes are complete, two separate operating systems exist, which each provide separate management interfaces to configure and manage each component.

The Native Mode Boot Process

With native mode, the following describes the files that each module boots from:

- **MSFC Boot Loader—**The MSFC-1 still requires the boot loader file in native mode (the MSFC-2 does not require this file). This file is identical to the boot loader file used in hybrid mode. If you have an MSFC-2, you don't need this file; however, it is recommended to keep this file so that you can revert back to hybrid mode.
- **Combined Supervisor and MSFC Image—**A single combined Supervisor and MSFC Cisco IOS operating system image is normally stored in the *bootflash:* device of the Supervisor and always begins with a prefix of c6sup, following by two digits that identify the Supervisor module and MSFC that the image is compatible for. For example, the file c6sup22-jsv-mz.121-11b.E4 is a native mode image for a Supervisor 2 module (indicated by the first number 2 digit) and MSFC-2 (indicated by the second number 2 digit).

When a native mode switch first boots, the switch processor on the Supervisor module reads a boot environment variable, which specifies the full path to the operating system image that should be booted. This image is a native mode (Cisco IOS) image, which boots the Supervisor module. The Supervisor module initializes the various modules installed, runs systems diagnostics, and loads the Cisco IOS operating system into Supervisor memory. Once complete, system control is then handed to the MSFC route processor, which boots from a portion of the native mode image. The MSFC hardware is initialized, system diagnostics are run and the Cisco IOS image is loaded into MSFC memory. At this point, both the switch processor and route processor have each loaded a separate Cisco IOS-based operating system that actually run independently of each other to some extent.

The MSFC Cisco IOS operating system has control over the console port on the Supervisor module and provides a single management interface that you as the administrator see from the switch console port, which allows you to perform all system configuration tasks. In the background the route processor handling the commands executed by administrators actually passes the tasks that require handling by the switch processor to the switch processor for execution. This serves to give the look and feel of a unified switch management interface that manages all components of the switch.

Configuration Prerequisites

Before beginning this scenario, it is important that you have an appropriate operating system file required for native mode operation and that this file is somehow distributed to a local file system on Switch-A. The easiest way of distributing the new native mode operating system image to Switch-A is used the network, which requires a TFTP server to be accessible from Switch-A. This scenario assumes that the appropriate native mode image has been obtained (if you are a registered cisco.com user and have sufficient rights to download software, you can obtain a native mode operating system image from http://www.cisco.com/cgi-bin/tablebuild.pl/cat6000-sup-ios) and that this image has been installed to a PCMCIA Flash card in Switch-A via TFTP.

Configuration Tasks

It is important to ensure you fully understand the native mode upgrade process and that you follow the process correctly to ensure a smooth upgrade to native mode. Upgrading from hybrid mode to native mode requires the following configuration tasks:

- Backing up existing configuration and operating system files
- Obtaining the appropriate native mode operating system files
- Upgrading to native mode operation

Backing up Existing Configuration and Operating System Files

The first thing that you must do before upgrading to native mode is to ensure that you have full backups of your current configuration and hybrid mode files. When upgrading to native mode a blank configuration is initially loaded, so you must manually reconfigure the switch to your previous configuration.

TIP

An automatic configuration converter is available on CCO at http://www.cisco.com/cgi-bin/Support/CatCfgConversion/catcfg_xlat.pl, which allows you to paste an existing CatOS hybrid mode configuration and outputs an equivalent native mode Cisco IOS configuration. This converter is available only for registered CCO users.

It is a good idea to also keep a copy of your hybrid mode files, just in case you need to revert back to hybrid mode. The ideal situation is to have enough flash so that you can leave the old hybrid mode image intact alongside the new native mode image. This makes it very quick and easy to revert back to hybrid mode if required.

At this stage, it is a good idea to have an understanding of the files that the Supervisor 2 engine and MSFC-2 on Switch-A are using for hybrid mode operation, so that you are aware of the files required to run hybrid mode if a roll back should be required. Example 6-25 demonstrates viewing the files used by the Supervisor engine on Switch-A and then establishing a console connection to the MSFC-2 on Switch-A by using the **session** command and viewing the files present on the MSFC-2 internal Flash file system.

Example 6-92 Viewing Hybrid Mode Files on an MSFC-2

```
Switch-A> (enable) dir bootflash:
-#- -length- -----date/time------ name
 1 6199068 Apr 26 2002 13:18:19 cat6000-sup2k8.7-2-2.bin

25782500 bytes available (6199068 bytes used)
Switch-A> (enable) session 15
Trying Router-15...
Connected to Router-15.
Escape character is '^]'.

Switch-A-MSFC> enable
Password: *****
Switch-A-MSFC# dir bootflash:
Directory of bootflash:/

  1 -rw-   1686724  Jun 04 2002 13:32:23 c6MSFC-2-boot-mz.121-8a.E4
  2 -rw-  12263928  Jun 04 2002 13:37:19 c6MSFC-2-jsv-mz.121-8a.E4

15204352 bytes total (1253444 bytes free)
```

In Example 6-25, you initially can see the CatOS operating system file used for the Supervisor 2 engine (cat6000-sup2k8.7-2-2.bin) present in the bootflash: file system on Switch-A (the internal Supervisor engine Flash). On the MSFC-2, you can see two files present in the bootflash: file system (not to be confused with the bootflash: file system on the Supervisor), which represents the internal Flash on the MSFC-2. The first file is the boot image (c6MSFC-2-boot-mz.121-8a.E4), which is required to initially boot the MSFC-2 when operating in hybrid mode. The second file is the actual operating system file (c6MSFC-2-jsv-mz.121-8a.E4) for the MSFC-2, which is designed to operate in hybrid mode.

With Switch-A running in native mode, the CatOS operating system file on the Supervisor bootflash: device is replaced with a single native mode Cisco IOS operating system image, which manages both the Supervisor 2 engine and MSFC-2 without any additional files. This means you don't actually need the boot

image and operating system image files located on the MSFC-2 bootflash: device. However, it is a good idea to maintain these files, at least during the upgrade process, just in case you need to revert to hybrid mode.

TIP

If you are upgrading a Supervisor 1 with MSFC-1 to native mode (instead of MSFC-2), you must keep the boot loader image in the MSFC flash file system, as it is used to boot the MSFC-1 in native mode.

Obtaining the Appropriate Native Mode Operating System Files

Before beginning the native mode upgrade process, you must ensure that you use the correct native mode image for your Catalyst 6000/6500 supervisor/MSFC configuration. A separate native mode image exists for the various combinations of Supervisor 1/2 modules and MSFC-1/MSFC-2 modules. For example, a different native mode image is required for a Supervisor 1 module with MSFC-2, compared with a Supervisor 2 module with MSFC-1. To determine which native mode image is suitable for your system, refer to the prefix of the native mode image filename. The first five characters of the filename should contain the text c6sup, which indicates the file is a native mode image. The next character (the sixth character of the filename) indicates the Supervisor engine that the native mode image is suitable for, while the following character (the seventh character of the filename) indicates the MSFC module that the native mode image is suitable for. For example, the prefix c6sup12 indicates the image is suitable for a Supervisor 1 with MSFC-2, while the prefix c6sup22 indicates the image is suitable for Supervisor 2 with MSFC-2.

If you have an MSFC-1 installed with the Supervisor 1, you must also ensure that the boot loader file used to initially boot the MSFC in hybrid mode is also present in native mode (i.e., the c6MSFC-2-boot-xxxx file). This file is already required for hybrid mode operation, so the correct boot loader file should already be in place.

After you have determined the appropriate files required for native mode, you must ensure the files are present on the Flash file system of the switch. Because the Supervisor module initially boots from the native mode image, the native mode image must be present on a file system that the Supervisor can initially read at system startup. This includes the following flash devices:

- **Supervisor internal Flash—**This is the internal Flash included with every Supervisor 2 module. This device is referred to by the Supervisor module as *bootflash:* and by the MSFC as *sup-bootflash:*.
- **PCMCIA Flash—**If your internal Flash does not have enough space to accommodate the native mode image, you must install PCMCIA-based Flash in the external PCMCIA slots on the Supervisor module. It is recommended that you have PCMCIA Flash available during the native mode upgrade, as you need to format all file systems after the upgrade to allow native mode to write to each file systems. PCMCIA Flash can be used as temporary storage whilst the internal Flash file system is being formatted.

In this scenario, assume that Switch-A has a PCMCIA flash card installed which allows for the native mode image to be stored here temporarily while the Supervisor Flash file system is formatted; this is required to ensure the file system is compatible with the native mode Cisco IOS operating system.

NOTE

A switch running in native mode can read a hybrid mode file system, but cannot write to the hybrid mode file system due to differences in the file system format. This is the reason why all file systems used by native mode must be reformatted.

Assuming a native mode image has been downloaded to the PCMCIA Flash card via TFTP or some other means, Example 6-26 demonstrates verifying a native mode image is present on the PCMCIA card.

Example 6-93 Verifying Native Mode Image is Available on File System

```
Switch-A> (enable) dir slot0:
-#- -length- -----date/time------ name
 1 21611516 Jun 03 2002 09:12:08 c6sup22-jsv-mz.121-11b.E4

10370052 bytes available (21611516 bytes used)
```

In Example 6-26, **slot0:** represents the Flash file system on the PCMCIA Flash card. An image called c6sup22-jsv-mz.121-11b.E4 is present. This is the new native mode image that you configure the switch to boot from. You can tell that the image is a native mode image for the Supervisor 2 with MSFC-2 because the filename prefix is c6sup22.

Upgrading to Native Mode Operation

Once the appropriate native mode files are in place within Flash, you are ready to begin the process of upgrading the switch to operate in native mode. Configuring the switch to operate in native mode requires the following configuration tasks:

- Boot into ROMMON mode
- Boot into native mode for the first time
- Convert Flash file systems to native mode format
- Set boot parameters
- Reboot the switch

Booting into ROMMON Mode

When you upgrade to native mode, you must alter the BOOT environment variables that the switch processor reads (stored in the ROMMON configuration) so that the switch processor boots the native mode image, rather than a hybrid mode CatOS image. You need to specify the correct file system path, so it is important you understand the Flash device upon which the Native mode image is stored, and the full name of the image. The full path includes the Native mode image filename, preceded by the Flash device name (e.g., bootflash:c6sup22-jsv-mz.121-11b.E4 refers to a native mode image installed on the internal Flash of the Supervisor).

NOTE

It is important to understand that you cannot use any file system device that is attached to the MSFC to store the native mode image because the Supervisor module must have access to the image upon initialization (remember the MSFC is not initialized until after the Supervisor has initially booted from the native mode image).

To reconfigure the boot environment variables, you need to initially boot the switch into ROMMON mode and then modify the boot environment variables. This can be achieved by modifying the boot variables of the configuration register on the Supervisor engine using the **set boot config-register** command, so that the switch always boots to ROMMON mode and then physically rebooting the switch. Example 6-27 demonstrates configuring Switch-A to always boot into ROMMON mode and then rebooting the switch into ROMMON mode.

Example 6-94 Booting the Catalyst 6000/6500 into ROMMON Mode

```
Switch-A> (enable) set boot config-register 0x0
Configuration register is 0x0
ignore-config: disabled
auto-config: non-recurring, overwrite, sync disabled
console baud: 9600
boot: the ROM monitor
Switch-A> (enable) reset
This command will reset the system.
Do you want to continue (y/n) [n]? y
2002 Jul 01 11:48:47 %SYS-5-SYS_RESET:System reset from Console
Powering OFF all existing linecards
System Bootstrap, Version 6.1(4)
Copyright 1994-2001 by cisco Systems, Inc.
c6k_sup2 processor with 131072 Kbytes of main memory

rommon 1 >
```

In Example 6-27, notice that the configuration register is set to **0x0** (short for 0x0000), which configures the switch to boot to ROMMON mode as indicated by the shaded output of the command. The switch is then rebooted using the **reset** command, with the switch booting into ROMMON mode due to the new value of the configuration register.

Booting into Native mode for the First Time

Before you can boot into native mode, you must clear a boot variable called CONFIG_FILE, which is used by hybrid mode to locate the configuration file that contains the configuration of the switch. The reason for this is that although native mode does not use this variable, there have been some issues with leaving this variable set a value other than null. Example 6-28 demonstrates clearing this variable in ROMMON mode on Switch-A.

Example 6-95 Clearing the CONFIG_FILE Environment Variable

```
rommon 1 > set
PS1=rommon ! >
BOOT=bootflash:cat6000-sup2k8.7-2-2.bin,1;
CONFIG_FILE=bootflash:switch.cfg
rommon 2 > CONFIG_FILE=
rommon 3 > sync
```

In Example 6-28, the **set** `ROMMON command is used to display the various environment variables—notice the CONFIG_FILE variable is currently set to` **bootflash:switch.cfg**, which represents a hidden CatOS binary configuration file stored in the internal flash file system (bootflash:) on Switch-A. This variable is then set to a null value using the command **CONFIG_FILE=**, after which the **sync** command must be used to write the environment variable changes permanently. At this point, for the environment variable change to take effect, the switch must once again be rebooted. The switch reboots back into ROMMON mode because the configuration register is still set to 0x0. Example 6-29 demonstrates rebooting Switch-A after clearing the CONFIG_FILE environment variable.

Example 6-96 Rebooting Switch-A To Clear the CONFIG_FILE Environment Variable

```
rommon 4 > reset

System Bootstrap, Version 6.1(4)
Copyright 1994-2001 by cisco Systems, Inc.
c6k_sup2 processor with 131072 Kbytes of main memory
```

```
rommon 1 > set
PS1=rommon ! >
BOOT=bootflash:cat6000-sup2k8.7-2-2.bin,1;
CONFIG_FILE=
```

In Example 6-29, notice that the CONFIG_FILE variable now has a null value, which ensures the switch doesn't boot with a value for this variable.

After clearing the CONFIG_FILE variable, the switch can now be booted from the native mode operating system image file. This can be achieved by using the **boot** ROMMON command, specifying the full path to the native mode image, which in this scenario is stored in PCMCIA Flash and can be referenced by the path **slot0:c6sup22-jsv-mz.121-11b.E4** (see Example 6-26). Example 6-30 demonstrates booting Switch-A in ROMMON mode from the native mode operating system image file.

Example 6-97 Booting Native Mode Manually from ROMMON Mode

```
rommon 2 > boot slot0:c6sup22-jsv-mz.121-11b.E4
Self decompressing the image : ##############################################
################################################################################
################################################################################
################################################################################
################################################################################
[OK]
Restricted Rights Legend
Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
 of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
 (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.
cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134-1706
Cisco Internetwork Operating System Software
IOS (tm) c6sup2_sp Software (c6sup2_sp-SPV-M), Version 12.1(11b)E4,
   EARLY DEPLOYMENT RELEASE SOFTWARE (fc1)
Synced to mainline version: 12.1(11b)
TAC:Home:Software:Ios General:CiscoIOSRoadmap:12.1
Copyright 1986-2001 by cisco Systems, Inc.
Compiled Wed 28-Mar-01 18:36 by hqluong
Image text-base: 0x30020980, data-base: 0x306B8000
Start as Primary processor
00:00:05: %SYS-3-LOGGER_FLUSHING: System pausing to ensure console
   debugging output.
00:00:03: Currently running ROMMON from S (Gold) region
00:00:05: %OIR-6-CONSOLE: Changing console ownership to route processor
System Bootstrap, Version 12.1(3r)E2, RELEASE SOFTWARE (fc1)
Copyright 2000 by cisco Systems, Inc.
Cat6k-MSFC2 platform with 131072 Kbytes of main memory
rommon 1 > boot
Self decompressing the image : ##############################################
################################################################################
## [OK]
Restricted Rights Legend
Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
 of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
 (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.
cisco Systems, Inc.
170 West Tasman Drive
```

```
San Jose, California 95134-1706
Cisco Internetwork Operating System Software
IOS (tm) MSFC2 Software (C6MSFC2-BOOT-M), Version 12.1(8a)E4,
   EARLY DEPLOYMENT RELEASE SOFTWARE (fc1)
Copyright 1986-2000 by cisco Systems, Inc.
Compiled Sat 14-Oct-00 05:33 by eaarmas
Image text-base: 0x30008980, data-base: 0x303B6000
cisco Cat6k-MSFC2 (R7000) processor with 114688K/16384K bytes of memory.
Processor board ID SAD04430J9K
R7000 CPU at 300Mhz, Implementation 39, Rev 2.1, 256KB L2, 1024KB L3 Cache
Last reset from power-on
X.25 software, Version 3.0.0.
509K bytes of non-volatile configuration memory.
16384K bytes of Flash internal SIMM (Sector size 512K).
Press RETURN to get started!

--- System Configuration Dialog ---
Continue with configuration dialog? [yes/no]: n
Router> enable
```

In Example 6-30, you can see the native mode boot process. First of all, the Supervisor engine is booted, as indicated in the first shaded line by the text, c6sup2 sp Software (Catalyst 6000 Supervisor 2 switch processor software). After the supervisor engine boots, notice the next shaded line, which states that console ownership is being changed to the route processor (MSFC):

```
00:00:05: %OIR-6-CONSOLE: Changing console ownership to route processor
```

This message indicates that the switch processor (Supervisor) is handing management control of the system to the route processor (MSFC). When a Catalyst 6000/6500 switch running native mode boots, the Supervisor initially executes startup code and then hands off the boot process to the MSFC. The MSFC then boots. At this stage, the MSFC still boots from the hybrid mode files located on the MSFC internal Flash. Once boot up by the MSFC is complete, notice that the switch loads with a blank configuration, as indicated by the System Configuration Dialog prompt. After specifying **n** at the System Configuration Dialog prompt, notice that a normal Cisco IOS **Router>** prompt is presented, as per the default operation when booting up a Cisco IOS-based Catalyst switch with a blank configuration.

At this point, it is a good idea to issue a **show version** command so that you can verify the switch has booted from the image you think it has booted from and to also verify that native mode has recognized each of the physical interfaces installed in the switch. Example 6-31 demonstrates using **show version** to verify the hardware configuration and operating system version on Switch-A.

Example 6-98 Verifying a Native Mode Catalyst 6000/6500 Switch

```
Router# show version
Cisco Internetwork Operating System Software
IOS (tm) c6sup2_rp Software (c6sup2_rp-JSV-M), Version 12.1(11b)E4,
   EARLY DEPLOYMENT
Synced to mainline version: 12.1(11b)
TAC:Home:Software:Ios General:CiscoIOSRoadmap:12.1
Copyright 1986-2001 by cisco Systems, Inc.
Compiled Wed 28-Mar-01 17:52 by hqluong
Image text-base: 0x30008980, data-base: 0x315D0000
ROM: System Bootstrap, Version 12.1(3r)E2, RELEASE SOFTWARE (fc1)
BOOTFLASH: c6sup2_rp Software (c6sup2_rp-JSV-M), Version 12.1(8a)EX,
   EARLY DEPLOYMENT
Router uptime is 2 hours, 33 minutes
System returned to ROM by power-on (SP by power-on)
Running default software
cisco Catalyst 6000 (R7000) processor with 114688K/16384K bytes of memory.
Processor board ID SAD04430J9K
```

```
R7000 CPU at 300Mhz, Implementation 39, Rev 2.1, 256KB L2, 1024KB L3 Cache
Last reset from power-on
Bridging software.
X.25 software, Version 3.0.0.
SuperLAT software (copyright 1990 by Meridian Technology Corp).
TN3270 Emulation software.
1 Virtual Ethernet/IEEE 802.3 interface(s)
48 FastEthernet/IEEE 802.3 interface(s)
2 Gigabit Ethernet/IEEE 802.3 interface(s)
381K bytes of non-volatile configuration memory.
16384K bytes of Flash internal SIMM (Sector size 512K).
Configuration register is 0x0
```

In Example 6-31, notice that the switch is running Cisco IOS 12.1(11b)E4 and that the switch has recognized 1 virtual Ethernet interface (the VLAN 1 interface), 48 FastEthernet interfaces (located in slot 2 of Switch-A), and 2 gigabit Ethernet interfaces, which are onboard the Supervisor 2 engine.

Converting the Flash File Systems to Native mode Format

Assuming the switch has booted correctly as shown in Example 6-30 and verified in Example 6-31, the switch is now operating in native mode which confirms the switch is compatible with the native mode image. The next step is to convert the existing Flash file systems to native mode format, so that native mode can write to these file systems. This involves the following tasks:

- Formatting existing hybrid mode file systems
- Copying native mode image to new native mode file systems

To format the Flash file systems, you use the **format** command which takes a single parameter indicated the Flash device that you wish to format. At a minimum, you must format the Flash from which the switch normally boots. This is normally the internal Flash on the Supervisor engine; however, it could also be a PCMCIA Flash card. In this scenario, although Switch-A is currently booted from the slot0: Flash device (i.e., a PCMCIA Flash card), this is only temporary with the intention of Switch-A normally booting from the internal Flash on the Supervisor engine. It is important to understand that in native mode, the MSFC refers to the internal Flash device on the Supervisor engine as **sup-bootflash:** (as the MSFC already has its own bootflash: internal flash device).

NOTE

It is recommended that all Flash file systems on the switch be formatted so that the switch can both read and write to all file systems.

Example 6-32 demonstrates formatting the onboard Flash file system of the Supervisor module and then verifying that the file system has been formatted.

Example 6-99 Formatting Supervisor Flash from Native Mode

```
Router# format sup-bootflash:
Format operation may take a while. Continue? [confirm] y
Format operation will destroy all data in "sup-bootflash:".
  Continue? [confirm] y
Formatting sector 1
Format of sup-bootflash complete
Router# dir sup-bootflash:
Directory of sup-bootflash:/

No files in directory
```

```
31981568 bytes total (31981568 bytes free)
```

In Example 6-32, after the format of the Supervisor internal Flash is complete, the **dir** command is used to verify that the file system is accessible.

NOTE

If you have an MSFC-2 installed, the native mode image will be larger than 16 MB. This means that your Supervisor Flash must be 32 MB to store the native mode image for the MSFC-2. If the onboard Flash is only 16 MB, you must store the native mode image on a 32-MB PCMCIA card permanently installed into the Supervisor engine.

After formatting the appropriate Flash file system that is used for booting the switch, you must next copy the native mode image to the newly formatted file system so that the switch boots. In this scenario, the Supervisor Flash (sup-bootflash:) is to be used to boot the switch, and the native mode image is currently stored on PCMCIA flash (slot0:). Example 6-33 demonstrates copying the native mode image on slot0: of Switch-A to the sup-bootflash: Flash device.

Example 6-100 Copying Native Mode Image to Onboard Flash

```
Router# copy slot0:c6sup22-jsv-mz.121-11b.E4 sup-bootflash:
c6sup22-jsv-mz.121-11b.E4
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
21611516 bytes copied in 371.077 secs (58340 bytes/sec)
Router# dir sup-bootflash:
Directory of sup-bootflash:/

  1 -rw-   21611516  July 01 2001 16:15:11 c6sup22-jsv-mz.121-11b.E4
31981568 bytes total (10370052 bytes free)
```

In Example 6-33, after the **copy** command is used for copying the native mode image, the **dir** command is then used to verify the image has been successfully copied to Flash.

NOTE

You can also copy the native mode image via other means such as over the network using TFTP or FTP; however, this requires some configuration of the switch because the switch loads with a blank configuration the first time you boot native IOS. Using the network is required when you have only a single Flash file system to boot the switch from initially (i.e., sup-bootflash:) because you must erase the native mode image when formatting the sup-bootflash: device in this situation.

At this stage, the native mode image is located in the appropriate Flash device, with the switch able to read and write from the file system because it is now formatted for native mode. It is a good idea at this point to also format all other Flash file systems for native mode operation, although you might want to leave this until such time that you are confident that the switch is operating correctly in native mode. If you format all other Flash file systems immediately, you lose the ability to rollback quickly to hybrid mode (you can still rollback; it just takes longer to do).

Set Boot Parameters

Finally, you need to modify the boot parameters so that the switch automatically boots from Flash and boots from the native mode image. At present, the configuration register of the Supervisor engine is set to boot to ROMMON mode, and the boot environment variable is set to the previous CatOS image file.

When working with boot variables in native mode, it is important to understand that just as in hybrid mode, the supervisor engine and MSFC each possess their own bootstrap code and associated boot variables. Although the MSFC controls the switch during normal operation, the switch initially boots from the Supervisor engine first, and then control is handed over to the MSFC. This means that you must ensure you modify the boot variables of both the Supervisor engine and MSFC.

To view the current boot environments on the MSFC, you can use the **show bootvar** command, as shown in Example 6-34.

Example 6-101 Verifying Boot Parameters on the MSFC in Native Mode

```
Router# show bootvar
BOOT variable = bootflash:c6MSFC-2-jsv-mz.121-8a.E4,1;
CONFIG_FILE variable does not exist
BOOTLDR variable = bootflash:c6MSFC-2-mz.121-8a.E4
Configuration register is 0x2102
```

In Example 6-34, the boot variables of the MSFC are actually the old boot variables of the MSFC in hybrid mode. This is quite reasonable because no modification of the MSFC environment variables has so far taken place. Also notice that the configuration register is 0x2102. This is a normal configuration register value that indicates the MSFC should boot normally from flash. The point of Example 6-34 is to show that the MSFC has separate boot environment variables from the Supervisor engine. For example, the configuration register of the Supervisor engine is currently 0x0 (boot to ROMMON mode), which is different from the configuration register on the MSFC.

To ensure both the Supervisor engine and MSFC boot correctly, you can set the boot environment variable from native mode using the **boot system** global configuration command. When executed, this command not only modifies the boot environment variables on the MSFC, it also modifies the boot environment variables on the Supervisor, as a single file is now used to boot both the Supervisor and MSFC. You must also modify the configuration register on the Supervisor engine so that the switch boots from Flash. This can be achieved by executing the **config-register** global configuration command in native mode. Example 6-35 demonstrates configuring the switch to boot from the new native mode image.

Example 6-102 Setting Native Mode Boot Parameters

```
Router# configure terminal
Router(config)# boot system sup-bootflash:c6sup22-jsv-mz.121-11b.E4
Router(config)# end
Router# copy running-config startup-config
Building configuration...
[OK]
Router# show bootvar
BOOT variable = sup-bootflash:c6sup22-jsv-mz.121-11b.E4,1;
CONFIG_FILE variable does not exist
BOOTLDR variable = bootflash:c6MSFC-2-mz.121-8a.E4
Configuration register is 0x2102
```

In Example 6-35, the **boot system** global configuration command is used to modify the BOOT environment variable to the native mode image path. The configuration is then saved, which ensures the environment variables are saved permanently. If you are using the MSFC-2 with native mode, the BOOTLDR variable is ignored. This is required only on the MSFC-1.

The configuration of Example 6-35 sets the boot environment variable for both the MSFC and the supervisor engine. To prove this, you can actually access the Supervisor engine switch processor from native mode and then view the boot environment variables for the Supervisor. To access the switch processor and view environment variables, you use the **remote login switch** command to access the switch processor and then use the **show bootvar** command to view the current boot parameters for the Supervisor engine itself, as shown in Example 6-36.

Example 6-103 Viewing the Supervisor Engine Boot Environment Variables

```
Router# remote login switch
Trying Switch ...
Entering CONSOLE for Switch
Type "^C^C^C" to end this session


Switch-sp# show bootvar
BOOT variable = bootflash:c6sup22-jsv-mz.121-11b.E4,12
CONFIG_FILE variable =
BOOTLDR variable does not exist
Configuration register is 0x0 (will be 0x2102 at next reload)
Switch-sp# exit

[Connection to Switch closed by foreign host]
Router#
```

You can see from Example 6-36 that native mode allows you to run Cisco IOS commands (not CatOS commands) from the Supervisor processor (switch processor). You shouldn't need to do this often, as commands executed on the MSFC in native mode are automatically passed to the appropriate switch processor or MSFC processor for handling. In Example 6-34, you need to use the switch processor to read the boot environment variables for the Supervisor engine rather than the MSFC. Notice that the BOOT variable is correctly set. This is because when you modify boot parameters in native mode, both the MSFC and Supervisor boot parameters are updated. The last shaded line indicates that the configuration register value on the Supervisor is 0x0, which means the switch still boots into ROMMON mode. You need to change this to ensure that the switch boots from Flash and loads the native mode image.

NOTE

The **remote command switch** privileged command can be used to execute commands on the switch processor from native mode, without having to first log in to the switch processor (see Example 6-37 for a demonstration).

To configure the switch to ensure it boots from Flash rather than into ROMMON mode as is currently the case, you can use the **config-register** global configuration mode command to modify the configuration register on both the MSFC and supervisor engine. Example 6-37 demonstrates modifying the configuration register and then using the **remote command switch** command to verify the new configuration register value on the Supervisor engine.

Example 6-104 Modifying the Supervisor Engine Configuration Register

```
Router# configure terminal
Router(config)# config-register 0x2102
Router(config)# end
Router# copy running-config startup-config
Building configuration...
[OK]
Router# remote command switch show bootvar

Switch-sp#
BOOT variable = bootflash:c6sup22-jsv-mz.121-11b.E4,1;
CONFIG_FILE variable =
BOOTLDR variable does not exist
Configuration register is 0x0 (will be 0x2102 at next reload)

Router#
```

In Example 6-37, after the configuration register is modified, the **remote command switch** command is used to execute the **show bootvar** command on the switch processor. Notice that the configuration register has been modified so that the Supervisor boots from Flash at the next reload (i.e., 0x2102).

Rebooting the Switch

Congratulations, you have successfully completed upgrading a Catalyst 6000/6500 switch with MSFC to native mode. At this stage, it is a good idea to reboot the switch to verify that the switch does boot automatically into native mode with no problems. After rebooting the switch, you can then configure the switch for native mode operation, which is discussed in the next scenario.

Scenario 6-4: Configuring CEF-Based Layer 3 Switching on the Catalyst 6000/6500 Operating in Native Mode

In the previous scenario, you learned how to upgrade a Cisco Catalyst 6000/6500 switch from hybrid mode to native mode. When you upgrade to native mode, you lose both the Supervisor engine and MSFC configurations, meaning you must manually reconfigure the functionality previously in place when the switch was in hybrid mode. In Scenario 6-2, you learned how to configure the Catalyst 6000/6500 for CEF-based Layer 3 switching in hybrid mode. In this scenario, you configure the newly converted Switch-A for CEF-based Layer 3 switching in native mode, maintaining the same functionality as in Scenario 6-2. Figure 6-16 shows the new topology of Switch-A for this scenario.

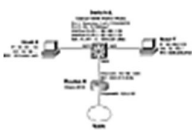


Figure 6-16 Topology for Scenario 6-4

Configuration Tasks

The goal of this scenario is to implement the same functionality configured on Switch-A in Scenario 6-2. When upgraded to native mode, a Cisco Catalyst 6000/6500 switch loses its configuration, meaning you must reconfigure the switch from scratch. This requires the following configuration tasks:

- Configuring system settings
- Configuring Layer 2 interfaces
- Configuring Layer 3 interfaces
- Configuring Layer 3 routing
- Verifying connectivity

Configuring System Settings

In this book so far, you have learned how to configure Cisco IOS-based Catalyst switches using the Catalyst 3550 switch. Because the native mode Catalyst 6000/6500 switch runs the same base operating system as the Catalyst 3550 (Cisco IOS), the commands used on the Catalyst 6000/6500 are identical to those used on the Catalyst 3550.

NOTE

Some minor differences exist in the command set supported on each switch, due to differences in the features supported.

This means that the same commands used to create and configure VLANs on the Catalyst 3550 are also used on the Catalyst 6000/6500. The same applies for all features that both switches have in common.

Example 6-38 demonstrates configuring Switch-A with a host name and passwords and configuring VLANs as required.

Example 6-105 Configuring System Settings on Switch-A in Native Mode

```
Router# configure terminal
Router(config)# hostname Switch-A
Switch-A(config)# enable secret cisco
Switch-A(config)# line vty 0 15
Switch-A(config-line)# password cisco
Switch-A(config-line)# exit
Switch-A(config)# vtp mode transparent
Setting device to VTP TRANSPARENT mode.
Switch-A(config)# vlan 2
Switch-A(config-vlan)# name VLAN02
```

Configuring Layer 2 Interfaces

On the native mode Catalyst 6000/6500, it is important to understand that by default, all interfaces are configured as routed interfaces, as opposed to switched interfaces as is the default on other Cisco IOS-based Catalyst switches. This means that all interfaces are shut down by default (this is the default state for a routed interface on Cisco IOS) so if you wish to configure an interface for Layer 2 or Layer 3 operation, you must explicitly enable the interface. If you want to configure a Layer 2 interface (i.e. switched interface), you must also explicitly configure this, because all interfaces are Layer 3 interfaces by default.

You might also be wondering how interfaces are named in native mode. All interfaces are named using the *interface-type interface-id* convention used on other Cisco IOS-based Catalyst switches, with the *interface-id* defined in the format *module/port*. For example, port 2/4 on a 10/100BASE-T module is represented as **interface fastEthernet2/4** on native mode.

In this scenario, interface fastEthernet0/1 through interface fastEthernet0/3 are all Layer 2 interfaces, with interface FastEthernet0/2 belonging to VLAN 2. Example 6-37 demonstrates configuring each of the above interfaces as Layer 2 interfaces and assigning the appropriate interface to VLAN 2.

Example 6-106 Configuring Layer 2 Parameters on Switch-A in Native Mode

```
Switch-A# configure terminal
Switch-A(config)# interface range fastEthernet2/1 - 3
Switch-A(config-if-range)# no shutdown
Switch-A(config-if-range)# switchport
Switch-A(config-if-range)# exit
```

```
Switch-A(config)# interface fastEthernet2/2
Switch-A(config-if)# switchport access vlan 2
```

In Example 6-39, notice that the **interface range** command is support on the native mode Catalyst 6000/6500, just like the Catalyst 3550. The **no shutdown** command takes each interface out of its default shutdown state and then the **switchport** command configures each interface as a Layer 2 interface, as opposed to a Layer 3 router interface. Finally, interface fastEthernet2/2 is assigned to VLAN 2, as this interface is connected to Host-Y.

Configuring Layer 3 Interfaces

On the native mode Catalyst 6000/6500, the same types of Layer 3 interfaces supported on other Cisco IOS-based Layer 3 Catalyst switches are also supported:

- Physical interface
- Switch virtual interface (SVI)

In this scenario, an SVI must be created for VLAN 2 (an SVI for VLAN 1 exists by default, but is in a shutdown state) so that Switch-A can route between VLAN 1 and VLAN 2. Example 6-40 demonstrates configuring SVIs for each VLAN and configuring the appropriate IP addressing for each SVI.

Example 6-107 Configuring Layer 2 Parameters on Switch-A in Native Mode

```
Switch-A# configure terminal
Switch-A(config)# interface vlan 1
Switch-A(config-if)# no shutdown
Switch-A(config-if)# ip address 192.168.1.1 255.255.255.0
Switch-A(config-if)# exit
Switch-A(config)# interface vlan 2
Switch-A(config-if)# ip address 192.168.2.1 255.255.255.0
```

As you can see in Example 6-40, the configuration required on Switch-A is identical to the configuration required on the MSFC when creating SVIs in hybrid mode (see Example 6-39).

Configuring Layer 3 Routing

Finally, Layer 3 routing needs to be configured. In this scenario, Switch-A needs to participate in OSPF area 0, with Router-A as an OSPF neighbor. Example 6-41 shows the configuration required on Switch-A.

Example 6-108 Configuring Layer 2 Parameters on Switch-A in Native Mode

```
Switch-A# configure terminal
Switch-A(config)# router ospf 1
Switch-A(config-router)# network 192.168.1.0 0.0.0.255 area 0
Switch-A(config-router)# network 192.168.2.0 0.0.0.255 area 0
Switch-A(config-router)# end
Switch-A# show ip ospf neighbors

Neighbor ID   Pri  State      Dead Time  Address     Interface
192.168.1.2    1  FULL/BDR    00:00:33  192.168.1.2   Vlan1
Switch-A# show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
    D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
    N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
    E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
    i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
    * - candidate default, U - per-user static route, o - ODR
    P - periodic downloaded static route
```

```
Gateway of last resort is not set

O IA 10.0.0.0/8 [110/2] via 192.168.1.2, 00:00:37, Vlan1
C  192.168.1.0/24 is directly connected, Vlan1
C  192.168.2.0/24 is directly connected, Vlan2
```

The configuration and subsequent Layer 3 routing configuration verification shown in Example 6-41 is identical to the hybrid mode configuration of an MSFC (see Example 6-40).

Verifying Connectivity

At this point, the native mode configuration of Switch-A is in place. Having a single operating system to manage all components of the Catalyst 6000/6500 simplifies management and ensures only a single configuration file and operating system image file needs to be maintained. To verify your configuration, Host-X should be able to ping Host-Y and vice versa, and both hosts should also be able to ping the loopback interface on Router-A.

To view the MAC address used by the MSFC for routing, you use the **show mls cef mac** command, which is demonstrated in Example 6-42 on Switch-A.

Example 6-109 Displaying the MSFC MAC Address in Native Mode

```
Switch-A# show mls cef mac

Switch-A-sp#
Router MAC address: 0004.dd41.890a
```

If you compare the output of Example 6-42 with the output of the **show mls cef mac** command on the Supervisor engine of Switch-A in Scenario 6-2 (see Example 6-41), notice that the same router MAC address (00d0.0515.640a) is displayed. Even though Switch-A is now operating in native mode, the fundamental components of Layer 3 switching are still present (PFC and MSFC), so it is important that the L2 engine passes any frames with a destination MAC address of the MSFC to the L3 engine for possible L3 switching. Notice in Example 6-42 that the **show mls cef mac** command is actually executed by the switch processor rather than the MSFC processor, as indicated by the **Switch-A-sp#** prompt at the beginning of the command output. This is because the L2 engine is a component of the switch processor (Supervisor engine).

To view the FIB table in native mode, you use the **show mls cef** command, as demonstrated in Example 6-43.

Example 6-110 Displaying the FIB in Native Mode

```
Switch-A# show mls cef

Switch-A-sp#
Index   Prefix      Mask        Adjacency
0     0.0.0.0     255.255.255.255   punt
1     255.255.255.255 255.255.255.255   punt
2     127.0.0.12    255.255.255.255   punt
3     127.0.0.0    255.255.255.255   punt
4     127.255.255.255 255.255.255.255   punt
5     192.168.1.1   255.255.255.255   0004.dd41.890a
6     192.168.1.2   255.255.255.255   00d0.0515.640a
7     192.168.1.100  255.255.255.255   0010.a4e0.1ed3
8     192.168.1.0   255.255.255.255   punt
9     192.168.1.255  255.255.255.255   punt
10     192.168.2.1   255.255.255.255  0004.dd41.890a
11     192.168.2.100  255.255.255.255   0006.53fe.8420
```

```
12     192.168.2.0   255.255.255.255   punt
13     192.168.2.255  255.255.255.255   punt
6400    224.0.0.0    255.255.255.0    punt
6401    192.168.1.0   255.255.255.0    punt
6402    192.168.2.0   255.255.255.0    punt
6403    10.0.0.0     255.0.0.0      00d0.0515.640a
115200   0.0.0.0     0.0.0.0       drop
```

In Example 6-43, notice that each destination prefix and associated mask are listed, along with adjacency information. The adjacency information does not specify next hop IP information as is the case in hybrid mode (see Example 6-42). Instead, the MAC address of the next hop device is listed. Notice that many prefixes have the punt adjacency associated with them. This means that any packets with a destination IP address that matches the prefix should be forwarded (punted) to the route processor (MSFC) for processing. For example, entry 5 shows that traffic destined to 192.168.1.1 should be sent to the route processor, which is the correct action because this is the IP address configured on the VLAN 1 interface. Notice that the last entry (numbered 115200) specifies a prefix of 0.0.0.0 and mask of 0.0.0.0, which matches any destination IP address not matched by the preceding FIB entries. The adjacency is a drop adjacency, which means any packets not matched by other FIB entries are dropped. This is the correct action to take because if no FIB entry for a destination exists, then no route for the destination exists and the destination is deemed unreachable.

Also in Example 6-43, the shaded entries indicate FIB entries that have a remote adjacency to which any packets that match the entry should be forwarded. The remote adjacency is indicated by a MAC address in the adjacency column. Referring to the MAC addresses for Host-X, Host-Y, and Router-A in Figure 6-16, you can see in Example 6-43 that the appropriate IP addresses are associated with the correct MAC addresses. For example, entry 6403 specifies a destination prefix of 10.0.0.0/8 (which is reachable via Router-A) and lists the adjacency as 00d0.0515.640a, which is the MAC address of Router-A.

Finally, to view the adjacency table in native mode, you use the **show mls cef adjacency** command, as demonstrated in Example 6-44.

Example 6-111 Displaying the CEF Adjacency Table in Native Mode

```
Switch-A# show mls cef adjacency

Switch-A-sp#
Index 17416 : mac-sa: 0004.dd41.890a, mac-da: 0006.53fe.8420
        interface: Fa2/2, mtu: 1514
        packets: 0000000000000010, bytes: 0000000000000960

Index 17417 : mac-sa: 0004.dd41.890a, mac-da: 00d0.0515.640a
        interface: Fa2/3, mtu: 1514
        packets: 0000000000000010, bytes: 0000000000000960

Index 17418 : mac-sa: 0004.dd41.890a, mac-da: 0010.a4e0.1ed3
        interface: Fa2/1, mtu: 1514
        packets: 0000000000000020, bytes: 0000000000001920

Index 262140: mac-sa: 00d0.0515.640a, mac-da: 0000.0000.0202
        interface: unknown, mtu: 1514
        packets: 0000004294967295, bytes: 0002199023255551
```

Notice in Example 6-44 that an adjacency entry exists per destination MAC address. The output is rather different from the comparative command on a hybrid mode switch and is much easier to understand. Each adjacency entry contains the information required for the hardware-based L3 engine on the PFC-2 to perform Layer 3 switching. Remember that Layer 3 switching occurs at the data plane of IP routing; at this level, a rewrite of the destination and source MAC addresses is required, along with knowledge of the egress interface. You can see in Example 6-44 that each adjacency specifies the source MAC address to rewrite,

destination MAC address to rewrite, and egress interface. You can also see the number of packets and associated bytes that have been matched to each adjacency. This is useful for verifying that CEF-based Layer 3 switching is actually taking place.

If you put together all of the information in Example 6-43 and Example 6-44, you begin to see exactly how L3 switching works using CEF. Suppose a packet from Host-X (192.168.1.100) is passed to the L3 engine on the PFC-2 that is addressed to Host-Y (192.168.2.100). The L3 engine inspects the destination IP address (192.168.2.100) and looks up the FIB table (see Example 6-44) for the most specific entry that matches the destination IP address. Entry #11 is matched, which specifies an adjacency of 0006.53fe.8420. The L3 engine then looks up the adjacency table (see Example 6-44) for an adjacency with a destination MAC address of 0006.53fe.8420. Entry #17416 is matched, which provides the L3 engine with the values for the rewriting of the source MAC address (0004.dd41.890a—i.e., the MSFC) and destination MAC address (0006.53fe.8420—i.e., Host-Y), as well as the interface (Fa2/2) out which the packet should be sent. The L3 engine rewrites the frame waiting to be routed, performs other required tasks (such as TTL decrement and checksum recomputation), and then forwards the packet to the egress interface for transmission. Figure 6-17 shows the process described above.

Figure 6-17 CEF Operation on Switch-A

At this stage, you have successfully verified CEF-based Layer 3 switching operation. As a final verification task, you can use the **show mls cef statistics** command to view the NetFlow statistical information that is collected. Example 6-45 demonstrates executing the **show mls cef statistics** command, generating some traffic on the network (in this example, a continuous ping is generated between Host-X and Host-Y), and then executing the **show mls cef statistics** command again.

Example 6-112 Displaying NetFlow Statistics

```
Switch-A# show mls cef statistics

Switch-A-sp#
Total CEF switched packets: 0000000000021720
Total CEF switched bytes:  0000000032444268
_ (Generate traffic between Host-X and Host-Y, wait a few minutes)
_
Switch-A# show mls cef statistics

Switch-A-sp#
Total CEF switched packets: 0000000000021926
Total CEF switched bytes:  0000000032753268
```

In Example 6-45, the second **show mls cef statistics** command is executed approximately 3 minutes after the first. Notice that the number of packets L3 switched by CEF has increased by 206 (from 21720 to 21926), and the total number of bytes has also increased due to the continuous ICMP traffic being sent from Host-X to Host-Y.

You can also use the **show interfaces** command on a native mode switch to view the number of packets that have been Layer 2 switched and Layer 3 switched on a particular interface. Example 6-46 shows the output of the **show interfaces** command on interface FastEthernet 2/2, which is connected to Host-Y.

Example 6-113 Displaying Interface Statistics in Native Mode

```
Switch-A# show interface FastEthernet2/2
FastEthernet2/2 is up, line protocol is up
 Hardware is C6k 10/100Mb 802.3, address is 0004.dd41.890a (bia 0004.dd41.890a)
_ (Output truncated)
```

```
_
 L2 Switched: ucast: 3993 pkt, 354635 bytes - mcast: 569 pkt, 46508 bytes
 L3 in Switched: ucast: 75329 pkt, 112930074 bytes - mcast: 0 pkt, 0 bytes mcast
 L3 out Switched: ucast: 75337 pkt, 112929372 bytes
_  (Output truncated)
_
```

In Example 6-46, the shaded output indicates how many packets have been L2 switched on the interface as well as how many packets have been L3 switched in and out of the interface. This command is useful for verifying L3 switching on a particular interface.

TIP

The L2 and L3 switching interface counters are updated approximately every 180 seconds.

Summary

The routing of IP traffic can be represented by two distinct planes of operation—the control plane and the data plane. The control plane deals with determining the information about where each destination IP address in the network is. Each destination IP address is associated with a next-hop IP address, which represents the next closest router to the destination. The data plane deals with the physical operations that are required to actually forward the packet to the next hop; this refers to the operation of storing the packet in memory while control plane information (such as the next hop address and egress interface) is determined, placing the appropriate Layer 2 addressing and encapsulation (depending on the egress interface Layer 2 protocol) on the frame, and then forwarding the packet out the appropriate egress interface to the next hop.

Layer 3 switching represents the data plane operations that are required to route IP traffic. Layer 3 switching can be performed in software, where an operating system application is responsible for the data plane operations, or in hardware, where a hardware chip (or ASIC) designed specifically for L3 switching is responsible for the data plane operations. Performing L3 switching in software allows for more flexibility because code can be written that uses as a common processor to perform the specific data plane operations required for each Layer 2 protocol. Performing L3 switching in hardware increases performance because all operations are performed by a function-specific chip, leaving the main processor free to perform other duties. Using hardware L3 switching is less flexible and more expensive because a hardware chip must be designed for each Layer 2 protocol (e.g., separate ASIC for Ethernet, separate ASIC for Token Ring).

Modern LAN design methodology recognizes the business requirements for the separation of functional groups within the LAN by using multiple VLANs. Separating functional groups increases network performance and efficiency and also allows for security access controls to be applied between each functional group (VLAN). From an IP perspective, each VLAN represents an IP subnet. For each IP subnet to communicate with remote IP subnets, IP routing or inter-VLAN routing is required. From a Layer 2 (Ethernet) perspective, IP routing is required between high-speed Ethernet networks which means that performance is a major consideration for inter-VLAN routing within the LAN. Hardware-based L3 switching within Ethernet LAN networks overcomes the performance limitations of software-based L3 switching and does not suffer the flexibility or cost disadvantages associated with hardware-based L3 switching, because only Ethernet-to-Ethernet L3 switching is required. This means only a single ASIC design is required.

Cisco Catalyst switches support two methods of hardware-based L3 switching. The methods differ in how the data plane components of L3 switching can get the necessary control plane information required to place in the Layer 2 framing. Multilayer switching (MLS) represents the first method of hardware-based L3 switching used by Cisco Catalyst switches and uses a flow-based model to populate a cache that includes the necessary control plane information for the data plane to L3 switch a packet. A flow simply represents a

collection of IP packets that each shares a number of identical parameters, such as the same destination IP address or same destination TCP port. An MLS Route Processor (MLS-RP) provides control plane operations, while an MLS Switching Engine (MLS-SE) provides data plane operation. MLS requires that the first packet of a new flow (candidate packet) received by the MLS-SE be routed to the MLS-RP, which makes a control plane decision and routes the packet in software to its destination. The MLS-SE sees the control plane information in the return packet (enabler packet) that is sent to the destination and populates the MLS cache with the necessary control plane information required to L3 switch packets that belong to the flow. Subsequent packets received by the MLS-SE can be L3 switched in hardware without requiring the packet to be sent to the MLS-RP because the MLS cache includes the required control plane information.

The next-generation method of hardware-based L3 switching is based upon Cisco Express Forwarding (CEF). In the CEF architecture, the cache (that is used to store the necessary control plane information required for the data plane hardware ASICs to L3 switch each packet) is pre-populated with all the necessary control plane information (the CEF table). This means that the L3 switching ASIC can switch all IP packets in hardware, unlike MLS which requires the first packet of a flow to be switched in software (by the MLS-RP). This architecture is more efficient and scalable and resolves performance limitations of MLS in environments where thousands of new flows are established every second. The CEF architecture is also very easy to scale because CEF caching information can be distributed to multiple L3 switching ASICs. This means that a L3 switch can perform multiple L3 switching operations simultaneously—one per CEF cache and ASIC. The route processor (control plane) component of IP routing is responsible for generating the information in the CEF table and updating it as routing topology changes occur. The CEF table actually consists of two tables—the Forwarding Information Base (FIB) and the adjacency table.

---

221 River Street, Hoboken, NJ 07030