

Access Layer



The access layer exists where the end users are connected to the network. Access switches usually provide Layer 2 (VLAN) connectivity between users. Devices in this layer, sometimes called building access switches, should have the following capabilities:

- Low cost per switch port
- High port density
- Scalable uplinks to higher layers
- High availability
- Ability to converge network services (that is, data, voice, video)
- Security features and quality of service (QoS)

Distribution Layer



The distribution layer provides interconnection between the campus network's access and core layers. Devices in this layer, sometimes called building *distribution switches*, should have the following capabilities:

- Aggregation of multiple access layer switches
- High Layer 3 routing throughput for packet handling
- Security and policy-based connectivity functions
- QoS features
- Scalable and redundant high-speed links to the core and access layers

In the distribution layer, uplinks from all access layer devices are aggregated, or come together. The distribution layer switches must be capable of processing the total volume of traffic from all the connected devices. These switches should have a high port density of high-speed links to support the collection of access layer switches.

VLANs and broadcast domains converge at the distribution layer, requiring routing, filtering, and security. The switches at this layer also must be capable of routing packets with high throughput.

Notice that the distribution layer usually is a Layer 3 boundary, where routing meets the VLANs of the access layer.

Core Layer



A campus network's core layer provides connectivity between all distribution layer devices. The core, sometimes referred to as the backbone, must be capable of switching traffic as efficiently as possible. Core switches should have the following attributes:

- Very high Layer 3 routing throughput
- No costly or unnecessary packet manipulations (access lists, packet filtering)

- Redundancy and resilience for high availability
- Advanced QoS functions

Devices in a campus network's core layer or backbone should be optimized for high-performance switching. Because the core layer must handle large amounts of campus-wide data, the core layer should be designed with simplicity and efficiency in mind.

Although campus network design is presented as a three-layer approach (access, distribution, and core layers), the hierarchy can be collapsed or simplified in certain cases. For example, small or medium-size campus networks might not have the size or volume requirements that would require the functions of all three layers. In that case, you could combine the distribution and core layers for simplicity and cost savings. When the distribution and core layers are combined into a single layer of switches, a *collapsed core* network results.

Modular Network Design

Designing a new network that has a hierarchy with three layers is fairly straightforward. You can also migrate an existing network into a hierarchical design. The resulting network is organized, efficient, and predictable. However, a simple hierarchical design does not address other best practices like redundancy, in the case where a switch or a link fails, or scalability, when large additions to the network need to be added.

Consider the hierarchical network shown in the left portion of Figure 1-8. Each layer of the network is connected to the adjacent layer by single links. If a link fails, a significant portion of the network will become isolated. In addition, the access layer switches are aggregated into a single distribution layer switch. If that switch fails, all the users will become isolated.

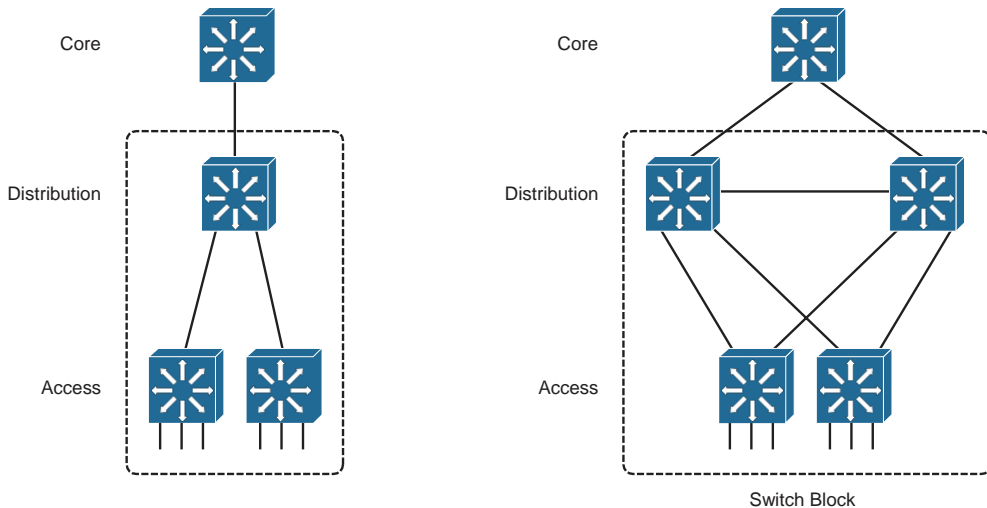


Figure 1-8 *Improving Availability in the Distribution and Access Layers*

To mitigate a potential distribution switch failure, you can add a second, redundant distribution switch. To mitigate a potential link failure, you can add redundant links from each access layer switch to each distribution switch. These improvements are shown on the right in Figure 1-8.

One weakness is still present in the redundant design of Figure 1-8: The core layer has only one switch. If that core switch fails, users in the access layer will still be able to communicate with each other. However, they will not be able to reach other areas of the network, such as a data center, the Internet, and so on. To mitigate the effects of a core switch failure, you can add a second, redundant core switch, as shown in Figure 1-9. Redundant links should also be added between each distribution layer switch and each core layer switch.

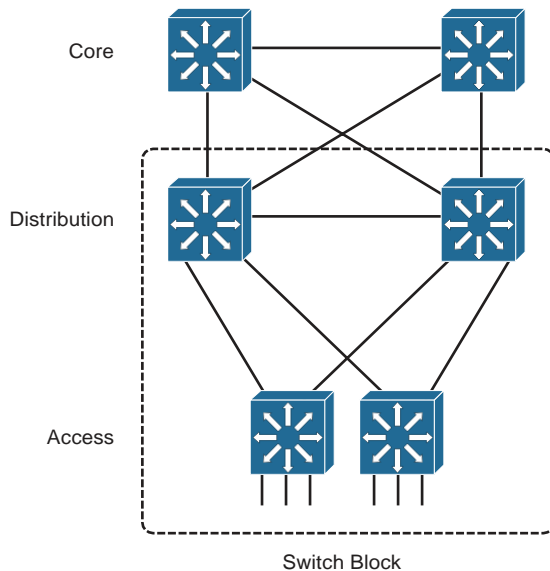


Figure 1-9 Fully Redundant Hierarchical Network Design

The redundancy needed for the small network shown in Figure 1-9 is fairly straightforward. As the network grows and more redundant switches and redundant links are added into the design, the design can become confusing. For example, suppose many more access layer switches need to be added to the network of Figure 1-9 because several departments of users have moved into the building or into an adjacent building. Should the new access layer switches be dual-connected into the same two distribution switches? Should new distribution switches be added, too? If so, should each of the distribution switches be connected to every other distribution *and* every other core switch, creating a fully meshed network?

Figure 1-10 shows one possible network design that might result. With so many interconnecting links between switches, it becomes a “brain-buster” exercise to figure out where VLANs are trunked, what the spanning-tree topologies look like, which links should have Layer 3 connectivity, and so on. Users might have connectivity through this network, but

it might not be clear how they are actually working or what has gone wrong if they are not working. This network looks more like a spider's web than an organized, streamlined design.

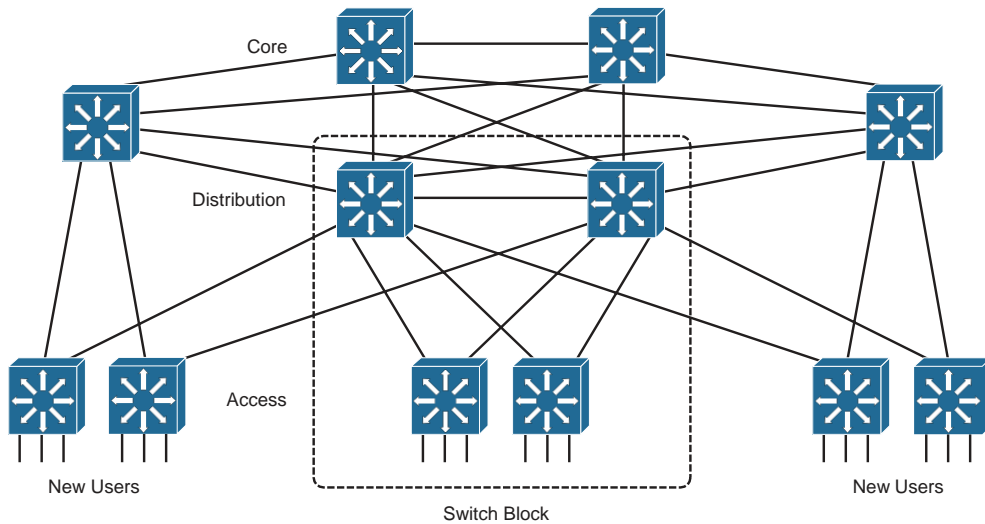


Figure 1-10 *Network Growth in a Disorganized Fashion*

To maintain organization, simplicity, and predictability, you can design a campus network in a logical manner, using a modular approach. In this approach, each layer of the hierarchical network model can be broken into basic functional units. These units, or modules, can then be sized appropriately and connected, while allowing for future scalability and expansion.

You can divide enterprise campus networks into the following basic elements or building blocks:

- **Switch block:** A group of access layer switches, together with their distribution switches. This is also called an *access distribution block*, named for the two switch layers that it contains. The dashed rectangle in Figures 1-8 through 1-10 represent typical switch blocks.
- **Core:** The campus network's backbone, which connects all switch blocks.



Other related elements can exist. Although these elements do not contribute to the campus network's overall function, they can be designed separately and added to the network design. For example, a data center containing enterprise resources or services can have its own access and distribution layer switches, forming a switch block that connects into the core layer. In fact, if the data center is very large, it might have its own core switches, too, which connect into the normal campus core. Recall how a campus network is divided into access, distribution, and core layers. The switch block contains switching devices from the access and distribution layers. The switch block then connects into the core layer, providing end-to-end connectivity across the campus. As the network grows, you can

add new access layer switches by connecting them into an existing pair of distribution switches, as shown in Figure 1-11. You could also add a completely new access distribution switch block that contains the areas of new growth, as shown in Figure 1-12.

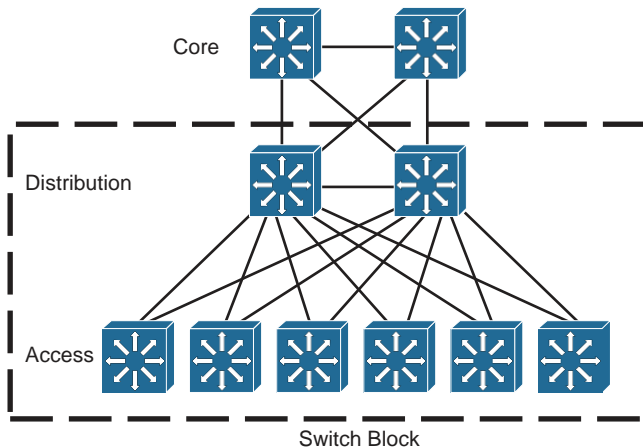


Figure 1-11 Network Growth by Adding Access Switches to a Switch Block

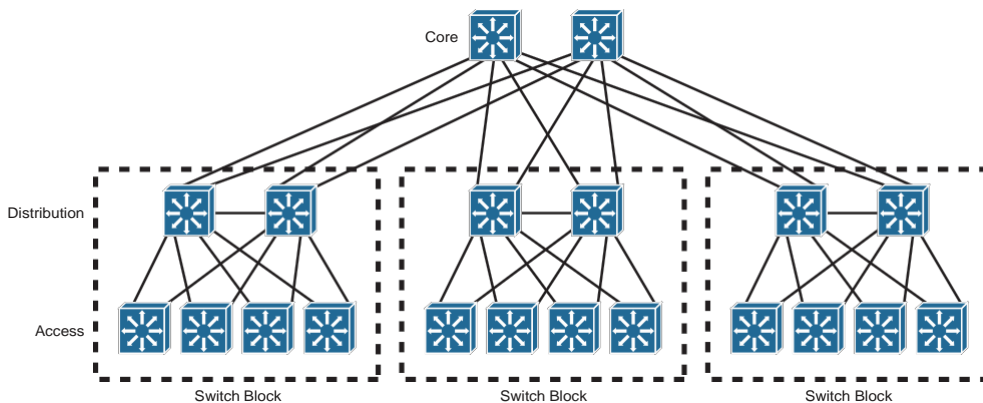


Figure 1-12 Network Growth by Adding New Switch Blocks

Sizing a Switch Block

Containing access and distribution layer devices, the switch block is simple in concept. You should consider several factors, however, to determine an appropriate size for the switch block. The range of available switch devices makes the switch block size very flexible. At the access layer, switch selection is usually based on port density or the number of connected users.

The distribution layer must be sized according to the number of access layer switches that are aggregated or brought into a distribution device. Consider the following factors:

- Traffic types and patterns
- Amount of Layer 3 switching capacity at the distribution layer
- Total number of users connected to the access layer switches
- Geographic boundaries of subnets or VLANs

Designing a switch block based solely on the number of users or stations contained within the block is usually inaccurate. Usually, no more than 2000 users should be placed within a single switch block. Although this is useful for initially estimating a switch block's size, this idea doesn't take into account the many dynamic processes that occur on a functioning network.

Instead, switch block size should be based primarily on the following:

- Traffic types and behavior
- Size and number of common workgroups

Because of the dynamic nature of networks, you can size a switch block too large to handle the load that is placed on it. Also, the number of users and applications on a network tends to grow over time. A provision to break up or downsize a switch block might be necessary as time passes. Again, base these decisions on the actual traffic flows and patterns present in the switch block. You can estimate, model, or measure these parameters with network-analysis applications and tools.

Note The actual network-analysis process is beyond the scope of this book. Traffic estimation, modeling, and measurement are complex procedures, each requiring its own dedicated analysis tool.

Generally, a switch block is too large if the following conditions are observed:

- The routers (multilayer switches) at the distribution layer become traffic bottlenecks. This congestion could be because of the volume of inter-VLAN traffic, intensive CPU processing, or switching times required by policy or security functions (access lists, queuing, and so on).
- Broadcast or multicast traffic slows the switches in the switch block. Broadcast and multicast traffic must be replicated and forwarded out many ports simultaneously. This process requires some overhead in the multilayer switch, which can become too great if significant traffic volumes are present.

Switch Block Redundancy

In any network design, the potential always exists for some component to fail. For example, if an electrical circuit breaker is tripped or shuts off, a switch might lose power. A better design is to use a switch that has two independent power supplies. Each power supply could be connected to two power sources so that one source is always likely to be available to power the switch. In a similar manner, a single switch might have an internal problem that causes it to fail. A single link might go down because a media module fails, a fiber-optic cable gets cut, and so on. To design a more resilient network, you can implement most of the components in redundant pairs.



A switch block consists of two distribution switches that aggregate one or more access layer switches. Each access layer switch should have a pair of uplinks—one connecting to each distribution switch. The physical cabling is easy to draw, but the logical connectivity is not always obvious. For example, Figure 1-13 shows a switch block that has a single VLAN A that spans multiple access switches. You might find this where there are several separate physical switch chassis in an access layer room, or where two nearby communications rooms share a common VLAN. Notice from the shading how the single VLAN spans across every switch (both access and distribution) and across every link connecting the switches. This is necessary for the VLAN to be present on both access switches and to have redundant uplinks for high availability.

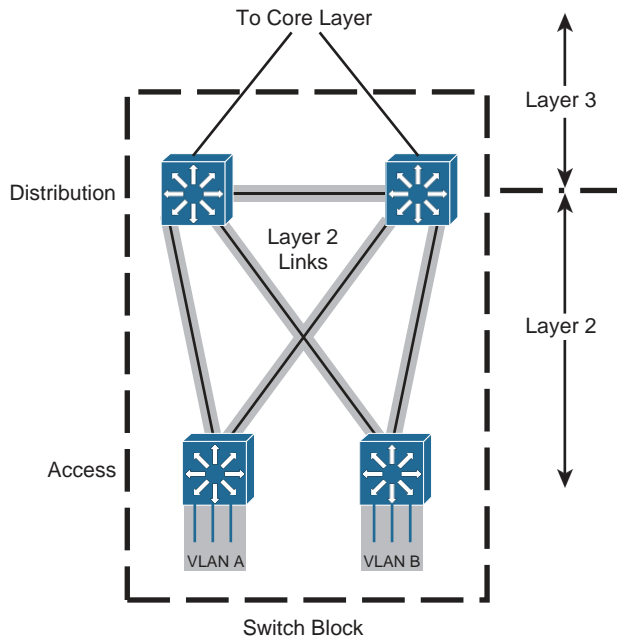


Figure 1-13 A Redundant Switch Block Design

Although this design works, it is not optimal. VLAN A must be carried over every possible link within the block to span both access switches. Both distribution switches must also support VLAN A because they provide the Layer 3 router function for all hosts on

the VLAN. The two distribution switches can use one of several redundant gateway protocols to provide an active IP gateway and a standby gateway at all times. These protocols require Layer 2 connectivity between the distribution switches and are discussed in Chapter 18, “Layer 3 High Availability.”

Notice how the shaded links connect to form two triangular loops. Layer 2 networks cannot remain stable or usable if loops are allowed to form, so some mechanism must be used to detect the loops and keep the topology loop free.

In addition, the looped topology makes the entire switch block a single failure domain. If a host in VLAN A misbehaves or generates a tremendous amount of broadcast traffic, all the switches and links in the switch block could be negatively impacted.

A better design works toward keeping the switch block inherently free of Layer 2 loops. As Figure 1-14 shows, a loop-free switch block requires a unique VLAN on each access switch. In other words, VLANs are not permitted to span across multiple access switches. The extent of each VLAN, as shown by the shaded areas, becomes a V shape rather than a closed triangular loop.

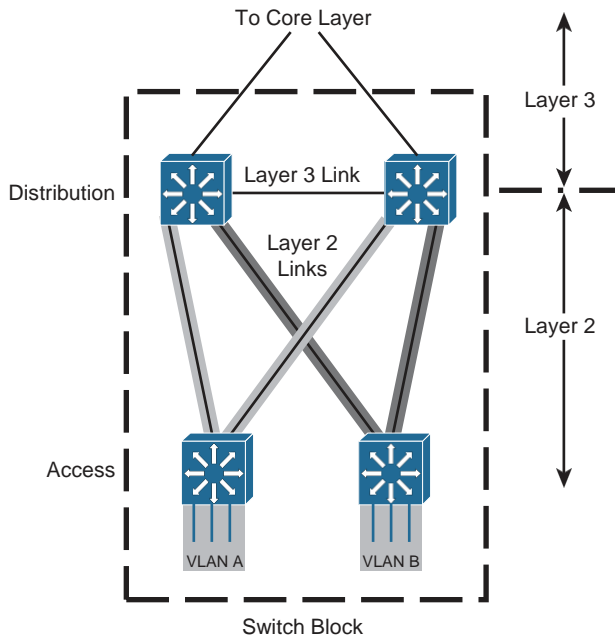


Figure 1-14 Best Practice Loop-Free Switch Block Topology



The boundary between Layers 2 and 3 remains the same. All Layer 2 connectivity is contained within the access layer, and the distribution layer has only Layer 3 links. Without any potential Layer 2 loops, the switch block can become much more stable and much less reliant on any mechanisms to detect and prevent loops. Also, because each access switch has two dedicated paths into the distribution layer, both links can be fully utilized with traffic load balanced across them. In turn, each Layer 3 distribution switch can load balance traffic over its redundant links into the core layer using routing protocols.

It is also possible to push the Layer 3 boundary from the distribution layer down into the access layer, as long as the access switches can support routing functions. Figure 1-15 illustrates this design. Because Layer 3 links are used throughout the switch block, network stability is offered through the fast convergence of routing protocols and updates. Routing can also load balance packets across the redundant uplinks, making full use of every available link between the network layers.

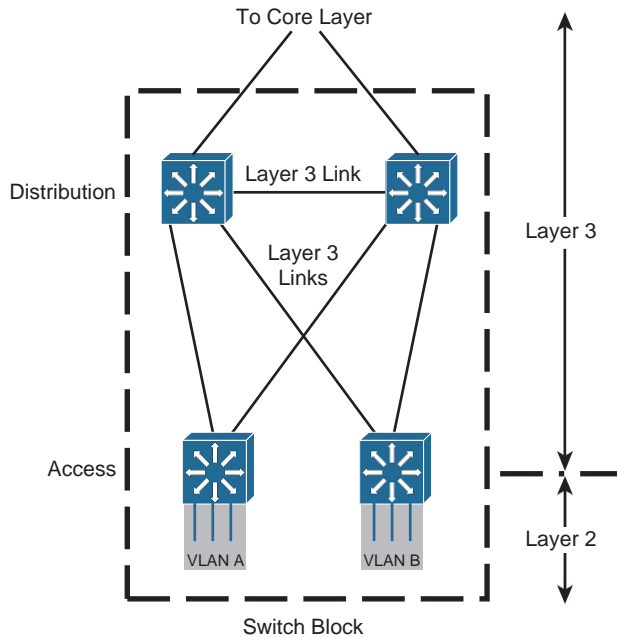


Figure 1-15 *A Completely Routed Switch Block*

You should become familiar with a few best practices that can help with a redundant hierarchical network design:

- Design each layer with pairs of switches.
- Connect each switch to the next higher layer with two links for redundancy.
- Connect each pair of distribution switches with a link, but do not connect the access layer switches to each other (unless the access switches support some other means to function as one logical stack or chassis).
- Do not extend VLANs beyond distribution switches. The distribution layer should always be the boundary of VLANs, subnets, and broadcasts. Although Layer 2 switches can extend VLANs to other switches and other layers of the hierarchy, this activity is discouraged. VLAN traffic should not traverse the network core.

Network Core

A core layer is required to connect two or more switch blocks in a campus network. Because all traffic passing to and from all switch blocks must cross the core, the core

layer must be as efficient and resilient as possible. The core is the campus network's basic foundation and carries much more traffic than any other switch block.

Recall that both the distribution and core layers provide Layer 3 functionality. Preferably, the links between distribution and core layer switches should be Layer 3 routed interfaces. You can also use Layer 2 links that carry a small VLAN bounded by the two switches. In the latter case, a Layer 3 switch virtual interface (SVI) is used to provide routing within each small VLAN.

The links between layers should be designed to carry the amount of traffic load handled by the distribution switches, at a minimum. The links between core switches should be of sufficient size to carry the aggregate amount of traffic coming into one of the core switches. Consider the average link utilization, but allow for future growth. An Ethernet core allows simple and scalable upgrades of magnitude; consider the progression from Gigabit Ethernet to 10-Gigabit Ethernet (10GE), and so on.



A core should consist of two multilayer switches that connect two or more switch blocks in a redundant fashion. A redundant core is sometimes called a *dual core* because it is usually built from two identical switches. Figure 1-16 illustrates the core. Notice that this core appears as an independent module and is not merged into any other block or layer.

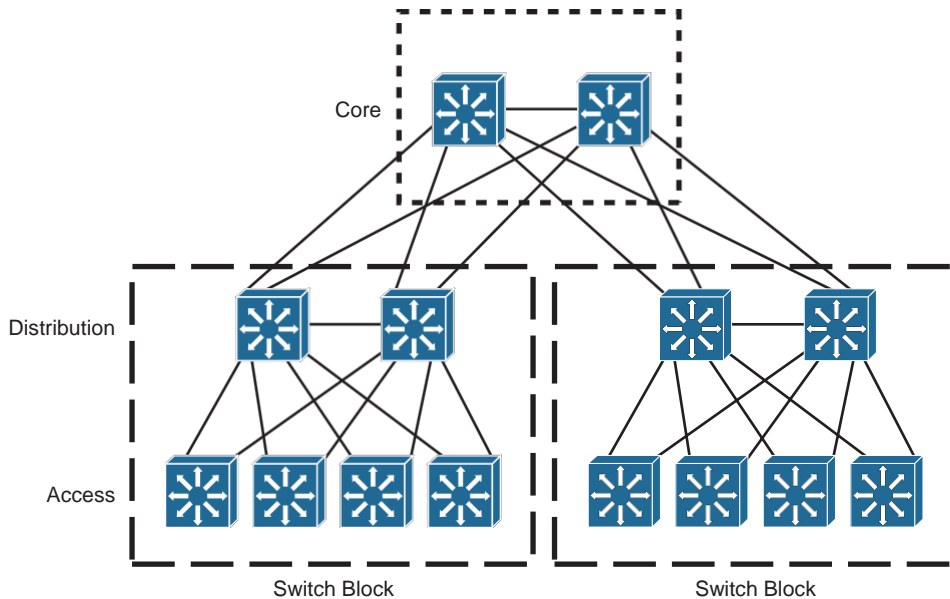


Figure 1-16 A Redundant Core Layer

Redundant links connect each switch block's distribution layer portion to each of the dual core switches. The two core switches connect by a common link.

With a redundant core, each distribution switch has two equal-cost paths into the core, allowing the available bandwidth of both paths to be used simultaneously. Both paths

remain active because the distribution and core layers use Layer 3 devices that can manage equal-cost paths in routing tables. The routing protocol in use determines the availability or loss of a neighboring Layer 3 device. If one switch fails, the routing protocol reroutes traffic using an alternative path through the remaining redundant switch.

If the campus network continues to grow to the point that it spans two large buildings or two large locations, the core layer can be replicated, as shown in Figure 1-17. Notice how the two-node redundant core has been expanded to include four core switches. This is known as a *multinode core*. Each of the four core switches is connected to the other core switches to form a fully meshed core layer.

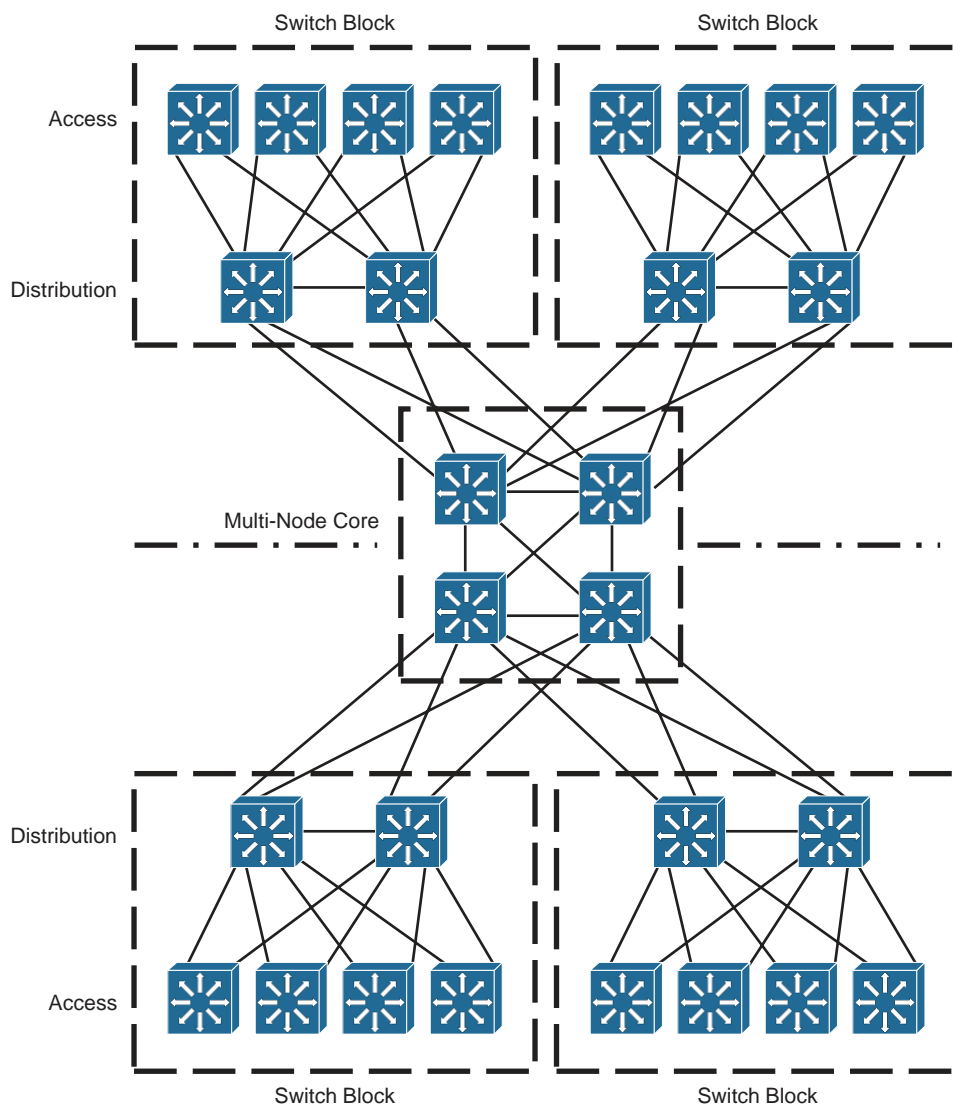


Figure 1-17 Using a Multi-Node Core in a Very Large Campus Network

Even though the multinode core is fully meshed, the campus network is still divided across the two pairs of core switches. Each switch block has redundant connections to only one core pair—not to all of the core switches.

Collapsed Core

Should all networks have a distinct redundant core layer? Perhaps not, in smaller campus networks, where the cost and scalability of a separate core layer is not warranted. A *collapsed core block* is one in which the hierarchy's core layer is collapsed into the distribution layer. Here, both distribution and core functions are provided within the same switch devices.

Figure 1-18 shows the basic collapsed core design. Although the distribution and core layer functions are performed in the same device, keeping these functions distinct and properly designed is important. Note also that the collapsed core is not an independent building block but is integrated into the distribution layer of the individual standalone switch blocks.

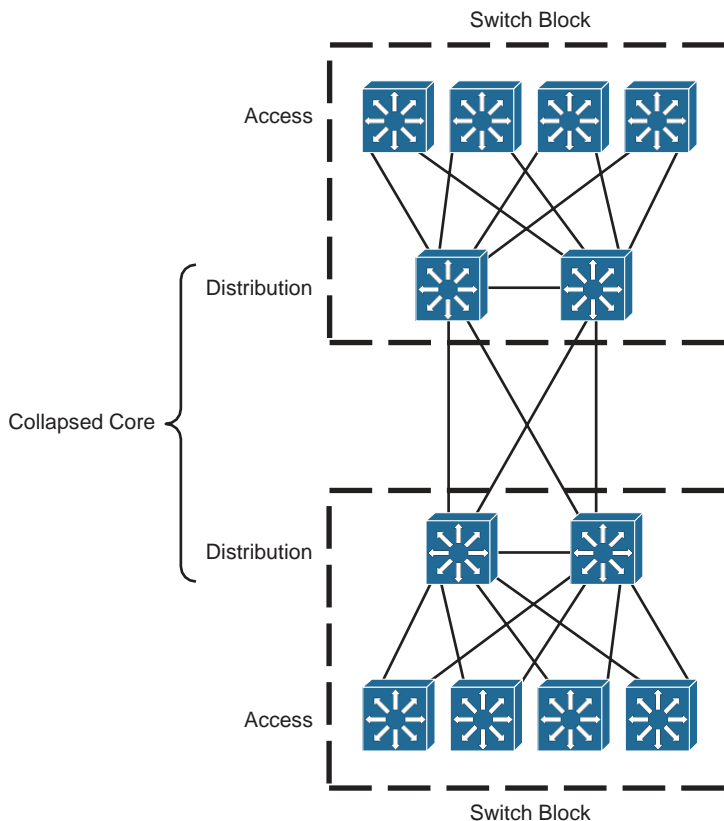


Figure 1-18 A Collapsed Core Network Design

In the collapsed core design, each access layer switch has a redundant link to each distribution layer switch. All Layer 3 subnets present in the access layer terminate at the distribution

switches' Layer 3 ports, as in the basic switch block design. The distribution switches connect to each other with redundant links, completing a path to use during a failure.

Core Size in a Campus Network

The core layer is made up of redundant switches and is bounded and isolated by Layer 3 devices. Routing protocols determine paths and maintain the core's operation. As with any network, you must pay some attention to the overall design of the routers and routing protocols in the network. Because routing protocols propagate updates throughout the network, network topologies might be undergoing change. The network's size (the number of routers) then affects routing protocol performance as updates are exchanged and network convergence takes place.

Although the network shown previously in Figure 1-16 might look small, with only two switch blocks of two Layer 3 switches (route processors within the distribution layer switches) each, large campus networks can have many switch blocks connected into the core. If you think of each multilayer switch as a router, you will recall that each route processor must communicate with and keep information about each of its directly connected peers. Most routing protocols have practical limits on the number of peer routers that can be directly connected on a point-to-point or multiaccess link. In a network with a large number of switch blocks, the number of connected routers can grow quite large. Should you be concerned about a core switch peering with too many distribution switches?

No, because the actual number of directly connected peers is quite small, regardless of the campus network size. Access layer VLANs terminate at the distribution layer switches (unless the access layer is configured for Layer 3 operation). The only peering routers at that boundary are pairs of distribution switches, each providing routing redundancy for each of the access layer VLAN subnets. At the distribution and core boundary, each distribution switch connects to only two core switches over Layer 3 switch interfaces. Therefore, only pairs of router peers are formed.

When multilayer switches are used in the distribution and core layers, the routing protocols running in both layers regard each pair of redundant links between layers as equal-cost paths. Traffic is routed across both links in a load-sharing fashion, utilizing the bandwidth of both.

One final core layer design point is to scale the core switches to match the incoming load. At a minimum, each core switch must handle switching each of its incoming distribution links at 100 percent capacity.

Cisco Products in a Hierarchical Network Design





Before delving into the design practices needed to build a hierarchical campus network, you should have some idea of the actual devices that you can place at each layer. Cisco has switching products tailored for layer functionality and for the size of the campus network.

For the purposes of this discussion, a large campus can be considered to span across many buildings. A medium campus might make use of one or several buildings, and a small campus might have only a single building.

Choose your Cisco products based on the functionality that is expected at each layer of a small, medium, or large campus. Do not get lost in the details of the tables. Rather, try to understand which switch fits into which layer for a given network size.




In the access layer, high port density, Power over Ethernet (PoE), and low cost are usually desirable. The Catalyst 2960-X, 3650, and 3850 switches provide 48 ports each. Like switch models can be connected to form a single logical switch when a greater number of ports is needed. The Catalyst 4500E is a single-switch chassis that can be populated with a variety of line cards. It also offers a choice of redundant supervisor modules that offer redundancy and even the ability to perform software upgrades with no impact to the production network. Table 1-3 describes some Cisco switch platforms that are commonly used in the access layer.

Table 1-3 *Common Access Layer Switch Platforms*

	Catalyst Model	Max Port Density	Uplinks	Max Backplane	Other Features
	2960-X	384 (Upto 8 48-port switches in a stack)	2 10GE or 4 1 Gigabit Ethernet per switch	80 Gbps	RIP, OSPF available for routed access layer; PoE+
	3650	432 (Upto 9 48-port switches in a stack)	2 Gigabit Ethernet or 4 10GE	160 Gbps	Full-featured routing available, integrated wireless controller, PoE+
	3850	432 (Upto 9 48-port switches in a stack)	4 Gigabit Ethernet, 4 10GE	480 Gbps	Full-featured routing available, integrated wireless controller, PoE+, UPoE
	4500E	384 (Up to 8 48-port modules per chassis)	Up to 12-port 10GE per module	928 Gbps	Dual supervisors, full-featured routing available, integrated wireless controller, PoE+, UPoE

The distribution and core layers are very similar in function and switching features. Generally, these layers require high Layer 3 switching throughput and a high density of high-bandwidth optical media. Cisco offers the Catalyst 3750-X, 4500-X, 4500E, and 6800, as summarized in Table 1-4.

Table 1-4 *Common Distribution and Core Layer Switch Platforms*

	Catalyst Model	Max Port Density	Max Backplane	Other Features
	4500-X	80 10GE	1.6 Tbps	Dual-chassis Virtual Switching System (VSS) redundancy
	4500E	96 10GE or 384 Gigabit Ethernet	928 Gbps	Dual supervisors
	6807-XL	4040Gbps, 160 Gigabit Ethernet, 480 Gigabit Ethernet	22.8 Tbps	Dual supervisor, dual-chassis VSS redundancy

Exam Preparation Tasks

Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the outer margin of the page. Table 1-5 lists a reference of these key topics and the page numbers on which each is found.



Table 1-5 *Key Topics for Chapter 1*

Key Topic Element	Description	Page Number
Paragraph	Describes the Cisco hierarchical network design principles	9
Paragraph	Describes the access layer	12
Paragraph	Describes the distribution layer	12
Paragraph	Describes the core layer	12
Paragraph	Explains modular network design using switch blocks	15
Paragraph	Discusses the pitfalls of letting VLANs span access layer switches	18
Paragraph	Discusses two best practice designs for switch block redundancy	19
Paragraph	Explains a redundant core design	21

Complete Tables and Lists from Memory

There are no memory tables in this chapter.

Define Key Terms

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

hierarchical network design, access layer, distribution layer, core layer, switch block, collapsed core, dual core



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

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

Example 9-1 *Detecting a Neighboring Switch’s STP Type*

```
Switch# show spanning-tree vlan 171
VLAN0171
  Spanning tree enabled protocol rstp
    Root ID      Priority      4267
                Address      00d0.0457.38aa
                Cost          3
                Port          833 (Port-channel1)
                Hello Time    2 sec Max Age 20 sec Forward Delay 15 sec

    Bridge ID     Priority      32939 (priority 32768 sys-id-ext 171)
                Address      0007.0d55.a800
                Hello Time    2 sec Max Age 20 sec Forward Delay 15 sec
                Aging Time    300

Interface      Role Sts Cost      Prio.Nbr Type
-----
Gil/0/7        Desg FWD 4         128.7    P2p
Gil/0/9/6      Altn BLK 4         128.9    P2p Peer(STP)
Po1            Root FWD 3         128.104  P2p
Po2            Desg FWD 3         128.834  P2p
Po3            Desg FWD 3         128.835  P2p
Switch#
```

The output in Example 9-1 shows information about the RSTP instance for VLAN 171. The first shaded line confirms that the local switch indeed is running RSTP. (The only other way to confirm the STP mode is to locate the **spanning-tree mode** command in the running configuration.)

In addition, this output displays all the active ports participating in the VLAN 171 instance of RSTP, along with their port types. The string **P2p** denotes a point-to-point RSTP port type in which a full-duplex link connects two neighboring switches that both are running RSTP. If you see *P2p Peer(STP)*, the port is a point-to-point type but the neighboring device is running traditional 802.1D STP.

Multiple Spanning Tree Protocol

Chapter 6 covered two “flavors” of spanning-tree implementations, IEEE 802.1Q and PVST+, both based on the 802.1D STP. These also represent the two extremes of STP operation in a network:

- **802.1Q:** Only a single instance of STP is used for all VLANs. If there are 500 VLANs, only 1 instance of STP will be running. This is called the *Common Spanning Tree* (CST) and operates over the trunk’s native VLAN.
- **PVST+:** One instance of STP is used for each active VLAN in the network. If there are 500 VLANs, 500 independent instances of STP will be running.

In most networks, each switch has a redundant path to another switch. For example, an access layer switch usually has two uplinks, each connecting to a different distribution or core layer switch. If 802.1Q's CST is used, only one STP instance will run. This means that there is only one loop-free topology at any given time and that only one of the two uplinks in the access layer switch will be forwarding. The other uplink always will be blocking.

Obviously, arranging the network so that both uplinks can be used simultaneously would be best. One uplink should carry one set of VLANs, whereas the other should carry a different set as a type of load balancing.

PVST+ seems more attractive to meet that goal because it allows different VLANs to have different topologies so that each uplink can be forwarding. But think of the consequences: As the number of VLANs increases, so does the number of independent STP instances. Each instance uses some amount of the switch CPU and memory resources. The more instances that are in use, the fewer CPU resources will be available for switching.

Beyond that, what is the real benefit of having 500 STP topologies for 500 VLANs, when only a small number of possible topologies exist for a switch with two uplinks? Figure 9-3 shows a typical network with an access layer switch connecting to a pair of core switches. Two VLANs are in use, with the root bridges configured to support load balancing across the two uplinks. The right portion of the figure shows every possible topology for VLANs A and B. Notice that because the access layer switch has only two uplinks, only two topologies actually matter—one in which the left uplink forwards, and one in which the right uplink forwards.

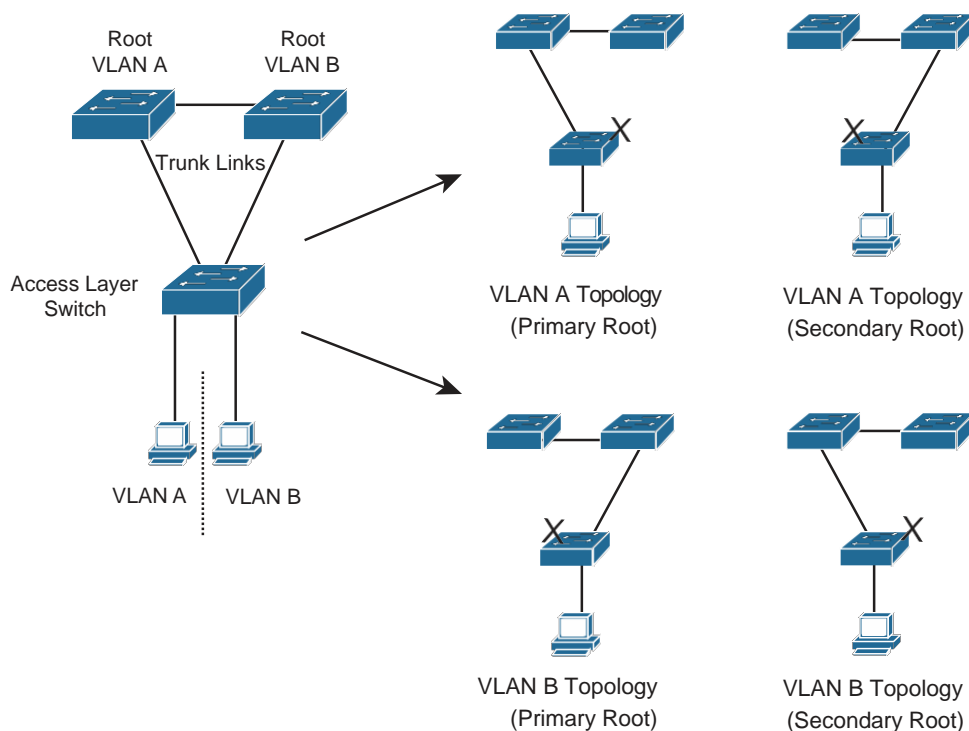


Figure 9-3 Possible STP Topologies for Two VLANs

Notice also that the number of useful topologies is independent of the number of VLANs. If 10 or 100 VLANs were used in the figure, there would still be only two possible outcomes at the access layer switch. Therefore, running 10 or 100 instances of STP when only a couple would suffice is rather wasteful.

The Multiple Spanning Tree Protocol was developed to address the lack of and surplus of STP instances. As a result, the network administrator can configure exactly the number of STP instances that makes sense for the enterprise network, no matter how many VLANs are in use. MST is defined in the IEEE 802.1s standard.

MST Overview

MST is built on the concept of mapping one or more VLANs to a single STP instance. Multiple instances of STP can be used (hence the name MST), with each instance supporting a different group of VLANs.

For the network shown in Figure 9-3, only two MST instances would be needed. Each could be tuned to result in a different topology so that Instance 1 would forward on the left uplink, whereas Instance 2 would forward on the right uplink. Therefore, VLAN A would be mapped to Instance 1, and VLAN B would be mapped to Instance 2.

To implement MST in a network, you need to determine the following:

- The number of STP instances needed to support the desired topologies
- Whether to map a set of VLANs to each instance

MST Regions



MST is different from 802.1Q and PVST+, although it can interoperate with them. If a switch is configured to use MST, it somehow must figure out which of its neighbors are using which type of STP. This is done by configuring switches into common MST regions, where every switch in a region runs MST with compatible parameters.

In most networks, a single MST region is sufficient, although you can configure more than one region. Within the region, all switches must run the instance of MST that is defined by the following attributes:

- MST configuration name (32 characters)
- MST configuration revision number (0 to 65535)
- MST instance-to-VLAN mapping table (4096 entries)

If two switches have the same set of attributes, they belong to the same MST region. If not, they belong to two independent regions.

MST BPDUs contain configuration attributes so that switches receiving BPDUs can compare them against their local MST configurations. If the attributes match, the STP instances within MST can be shared as part of the same region. If not, a switch is seen

to be at the MST region boundary, where one region meets another or one region meets traditional 802.1D STP.

Note The entire MST instance-to-VLAN mapping table is not sent in the BPDUs because the instance mappings must be configured on each switch. Instead, a digest, or a hash code computed from the table contents, is sent. As the contents of the table change, the digest value will be different. Therefore, a switch quickly can compare a received digest to its own to see if the advertised table is the same.

Spanning-Tree Instances Within MST

MST was designed to interoperate with all other forms of STP. Therefore, it also must support STP instances from each STP type. This is where MST can get confusing. Think of the entire enterprise network as having a single CST topology so that one instance of STP represents any and all VLANs and MST regions present. The CST maintains a common loop-free topology while integrating all forms of STP that might be in use.

To do this, CST must regard each MST region as a single “black box” bridge because it has no idea what is inside the region, nor does it care. CST maintains a loop-free topology only with the links that connect the regions to each other and to standalone switches running 802.1Q CST.

IST Instances



Something other than CST must work out a loop-free topology inside each MST region. Within a single MST region, an Internal Spanning Tree (IST) instance runs to work out a loop-free topology between the links where CST meets the region boundary and all switches inside the region. Think of the IST instance as a locally significant CST, bounded by the edges of the region.

The IST presents the entire region as a single virtual bridge to the CST outside. BPDUs are exchanged at the region boundary only over the native VLAN of trunks, as if a single CST were in operation. And, indeed, it is.

Figure 9-4 shows the basic concept behind the IST instance. The network at the left has an MST region, where several switches are running compatible MST configurations. Another switch is outside the region because it is running only the CST from 802.1Q.

The same network is shown at the right, where the IST has produced a loop-free topology for the network inside the region. The IST makes the internal network look like a single bridge (the “big switch” in the cloud) that can interface with the CST running outside the region.

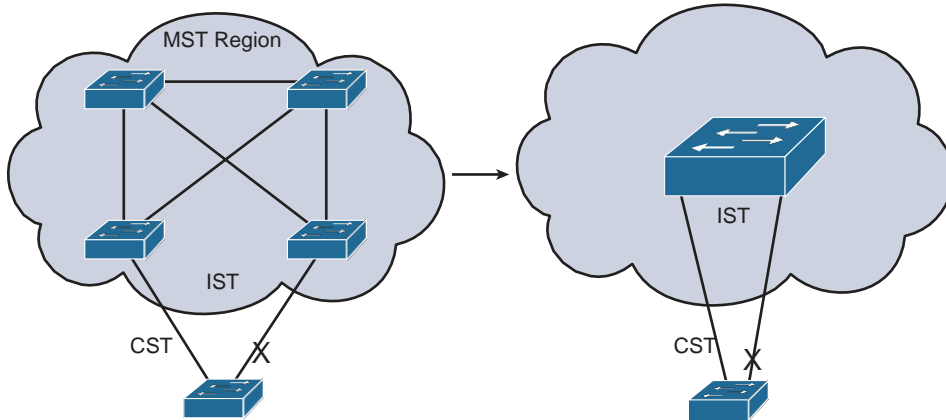


Figure 9-4 Concepts Behind the IST Instance

MST Instances



Recall that the whole idea behind MST is the capability to map multiple VLANs to a smaller number of STP instances. Inside a region, the actual MST instances (MSTI) exist alongside the IST. Cisco supports a maximum of 16 MSTIs in each region. The IST always exists as MSTI number 0, leaving MSTIs 1 through 15 available for use.

Figure 9-5 shows how different MSTIs can exist within a single MST region. The left portion of the figure is identical to that of Figure 9-4. In this network, two MST instances, MSTI 1 and MSTI 2, are configured with different VLANs mapped to each. Their topologies follow the same structure as the network on the left side of the figure, but each has converged differently.

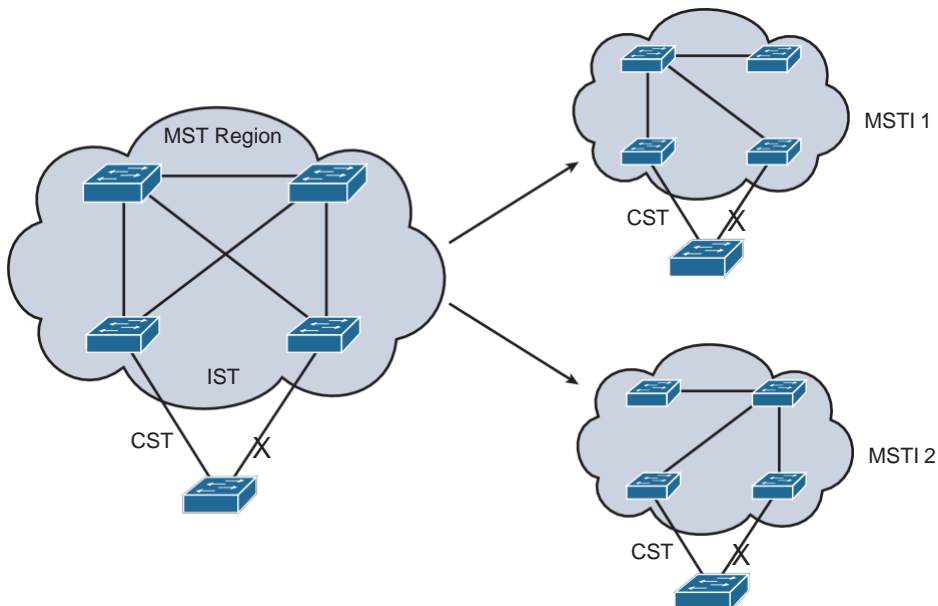


Figure 9-5 Concepts Behind MST Instances

Notice that within the MST cloud, there are now three independent STP instances coexisting: MSTI1, MSTI 2, and the IST.

Only the IST (MSTI 0) is allowed to send and receive MST BPDUs. Information about each of the other MSTIs is appended to the MST BPDU as an M-record. Therefore, even if a region has all 16 instances active, only 1 BPDU is needed to convey STP information about them all.

Each of the MSTIs is significant only within a region, even if an adjacent region has the same MSTIs in use. In other words, the MSTIs combine with the IST only at the region boundary to form a subtree of the CST. That means only IST BPDUs are sent into and out of a region.

What if an MST region connects with a switch running traditional PVST+? MST can detect this situation by listening to the received BPDUs. If BPDUs are heard from more than one VLAN (the CST), PVST+ must be in use. When the MST region sends a BPDU toward the PVST+ switch, the IST BPDUs are replicated into all the VLANs on the PVST+ switch trunk.

Tip Keep in mind that the IST instance is active on every port on a switch. Even if a port does not carry VLANs that have been mapped to the IST, IST must be running on the port. Also, by default, all VLANs are mapped to the IST instance. You must explicitly map them to other instances, if needed.

MST Configuration



You must manually configure the MST configuration attributes on each switch in a region. There is currently no method to propagate this information from one switch to another, as is done with a protocol such as VLAN Trunking Protocol (VTP). To define the MST region, use the following configuration commands in the order shown:

Step 1. Enable MST on the switch:

```
Switch(config)# spanning-tree mode mst
```

Step 2. Enter the MST configuration mode:

```
Switch(config)# spanning-tree mst configuration
```

Step 3. Assign a region configuration name (up to 32 characters):

```
Switch(config-mst)# name name
```

Step 4. Assign a region configuration revision number (0 to 65,535):

```
Switch(config-mst)# revision version
```

The configuration revision number gives you a means of tracking changes to the MST region configuration. Each time you make changes to the configuration, you should increase the number by one. Remember that the region configuration (including the revision number) must match on all switches in the

region. Therefore, you also need to update the revision numbers on the other switches to match.

Step 5. Map VLANs to an MST instance:

```
Switch(config-mst)# instance instance-id vlan vlan-list
```

The *instance-id* (0 to 15) carries topology information for the VLANs listed in *vlan-list*. The list can contain one or more VLANs separated by commas. You also can add a range of VLANs to the list by separating numbers with a hyphen. VLAN numbers can range from 1 to 4094. (Remember that, by default, all VLANs are mapped to instance 0, the IST.)

Step 6. Show the pending changes you have made:

```
Switch(config-mst)# show pending
```

Step 7. Exit the MST configuration mode; commit the changes to the active MST region configuration:

```
Switch(config-mst)# exit
```

After MST is enabled and configured, PVST+ operation stops and the switch changes to RSTP operation. A switch cannot run both MST and PVST+ at the same time.

You also can tune the parameters that MST uses when it interacts with CST or traditional 802.1D. The parameters and timers are identical to those discussed in Chapter 7, “Spanning-Tree Configuration.” In fact, the commands are very similar except for the addition of the **mst** keyword and the *instance-id*. Instead of tuning STP for a VLAN instance, you use an MST instance.

Table 9-2 summarizes the commands as a quick reference. Notice that the timer configurations are applied to MST as a whole, not to a specific MST instance. This is because all instance timers are defined through the IST instance and BPDUs.

Table 9-2 *MST Configuration Commands*

Task	Command Syntax
Set root bridge (macro).	Switch(config)# spanning-tree mst instance-id root {primary secondary} [diameter diameter]
Set bridge priority.	Switch(config)# spanning-tree mst instance-id priority bridge-priority
Set port cost.	Switch(config)# spanning-tree mst instance-id cost cost
Set port priority.	Switch(config)# spanning-tree mst instance-id port-priority port-priority
Set STP timers.	Switch(config)# spanning-tree mst hello-time seconds
	Switch(config)# spanning-tree mst forward-time seconds
	Switch(config)# spanning-tree mst max-age seconds

Aggregating Switch Links

In previous chapters, you learned about connecting switches and organizing users and devices into common workgroups. Using these principles, end users can be given effective access to resources both on and off the campus network. However, today’s mission-critical applications and services demand networks that provide high availability and reliability.

This chapter presents technologies that you can use in a campus network to provide higher bandwidth and reliability between switches.

“Do I Know This Already?” Quiz

The “Do I Know This Already?” quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the “Exam Preparation Tasks” section. If you are in doubt based on your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. Table 10-1 outlines the major headings in this chapter and the “Do I Know This Already?” quiz questions that go with them. You can find the answers in Appendix A, “Answers to the ‘Do I Know This Already?’ Quizzes.”

Table 10-1 “Do I Know This Already?” Foundation Topics Section-to-Question Mapping

Foundation Topics	Section	Questions Covered in This Section
Switch Port Aggregation with EtherChannel		1–7
EtherChannel Negotiation Protocols		8–10
EtherChannel Configuration		11–12
Troubleshooting an EtherChannel		13

1. If Gigabit Ethernet ports are bundled into an EtherChannel, what is the maximum throughput supported on a Catalyst switch?
 - a. 1 Gbps
 - b. 2 Gbps
 - c. 4 Gbps
 - d. 8 Gbps
 - e. 16 Gbps

- 2.** Which of these methods distributes traffic over an EtherChannel?
 - a.** Round robin
 - b.** Least-used link
 - c.** A function of address
 - d.** A function of packet size
- 3.** What type of interface represents an EtherChannel as a whole?
 - a.** Channel
 - b.** Port
 - c.** Port channel
 - d.** Channel port
- 4.** Which of the following is not a valid method for EtherChannel load balancing?
 - a.** Source MAC address
 - b.** Source and destination MAC addresses
 - c.** Source IP address
 - d.** IP precedence
 - e.** UDP/TCP port
- 5.** How can the EtherChannel load-balancing method be set?
 - a.** Per switch port
 - b.** Per EtherChannel
 - c.** Globally per switch
 - d.** Cannot be configured
- 6.** What logical operation is performed to calculate EtherChannel load balancing as a function of two addresses?
 - a.** OR
 - b.** AND
 - c.** XOR
 - d.** NOR
- 7.** Which one of the following is a valid combination of ports for an EtherChannel?
 - a.** Two access links (one VLAN 5, one VLAN 5)
 - b.** Two access links (one VLAN 1, one VLAN 10)
 - c.** Two trunk links (one VLANs 1 to 10, one VLANs 1,11 to 20)
 - d.** Two 10/100/1000 Ethernet links (both full duplex, one 100 Mbps)

8. Which of these is a method for negotiating an EtherChannel?
 - a. PAP
 - b. CHAP
 - c. LAPD
 - d. LACP
9. Which of the following is a valid EtherChannel negotiation mode combination between two switches?
 - a. PAgP auto, PAgP auto
 - b. PAgP auto, PAgP desirable
 - c. on, PAgP auto
 - d. LACP passive, LACP passive
10. When is PAgP's "desirable silent" mode useful?
 - a. When the switch should not send PAgP frames
 - b. When the switch should not form an EtherChannel
 - c. When the switch should not expect to receive PAgP frames
 - d. When the switch is using LACP mode
11. Which of the following EtherChannel modes does not send or receive any negotiation frames?
 - a. **channel-group 1 mode passive**
 - b. **channel-group 1 mode active**
 - c. **channel-group 1 mode on**
 - d. **channel-group 1 mode desirable**
 - e. **channel-group 1 mode auto**
12. Two computers are the only hosts sending IP data across an EtherChannel between two switches. Several different applications are being used between them. Which of these load-balancing methods would be more likely to use the most links in the EtherChannel?
 - a. Source and destination MAC addresses.
 - b. Source and destination IP addresses.
 - c. Source and destination TCP/UDP ports.
 - d. None of the other answers is correct.

- 13.** Which command enables you to see the status of an EtherChannel's links?
- a.** `show channel link`
 - b.** `show etherchannel status`
 - c.** `show etherchannel summary`
 - d.** `show ether channel status`

Foundation Topics

Switch Port Aggregation with EtherChannel

As discussed in Chapter 3, “Switch Port Configuration,” switches can use Ethernet, Fast Ethernet, Gigabit, or 10-Gigabit Ethernet ports to scale link speeds by a factor of 10. It might seem logical to simply add more links between two switches to scale the bandwidth incrementally. Suppose two switches have a single Gigabit Ethernet link between them. If you add a second link, will the available bandwidth double? No, because each link acts independently, a bridging loop could easily form through them. As the left portion of Figure 10-1 shows, STP will detect the loop potential and will place one of the links in the blocking state. The end result is still a single active link between switches. Even if you add several more links, STP will keep all but one in the blocking state, as shown on the right portion of Figure 10-1.

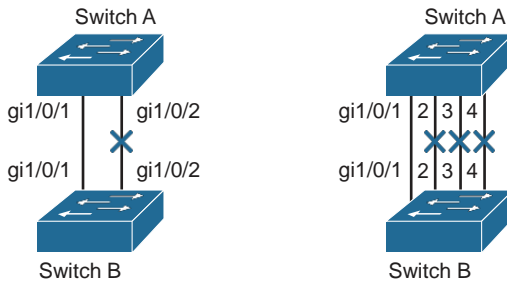


Figure 10-1 *The Effects of Trying to Scale Bandwidth with Individual Links*

Cisco offers another method of scaling link bandwidth by aggregating, or *bundling*, parallel links, termed the *EtherChannel* technology. Two to eight links of either Fast Ethernet (FE), Gigabit Ethernet (GE), or 10-Gigabit Ethernet (10GE) can be bundled as one logical link of Fast EtherChannel (FEC), Gigabit EtherChannel (GEC), or 10-Gigabit Etherchannel (10GEC), respectively. This bundle provides a full-duplex bandwidth of up to 1600 Mbps (eight links of Fast Ethernet), 16 Gbps (eight links of GE), or 160 Gbps (eight links of 10GE).

This also provides an easy means to “grow,” or expand, a link’s capacity between two switches, without having to continually purchase hardware for the next magnitude of throughput. For example, a single Fast Ethernet link (200 Mbps throughput) can be incrementally expanded up to eight Fast Ethernet links (1600 Mbps) as a single Fast EtherChannel. If the traffic load grows beyond that, the growth process can begin again with a single GE link (2 Gbps throughput), which can be expanded up to eight GE links as a Gigabit EtherChannel (16 Gbps). The process repeats again by moving to a single 10GE link, and so on.

Ordinarily, having multiple or parallel links between switches creates the possibility of bridging loops, an undesirable condition. EtherChannel avoids this situation by bundling parallel links into a single, logical link, which can act as either an access or a trunk link.

Switches or devices on each end of the EtherChannel link must understand and use the EtherChannel technology for proper operation. Figure 10-2 demonstrates how the links added in Figure 10-1 can be configured as an EtherChannel bundle. All the bundled physical links are collectively known by the logical EtherChannel interface, port channel 1. Notice that none of the physical links are in the Blocking state; STP is aware of the single port channel interface, which is kept in the Forwarding state.

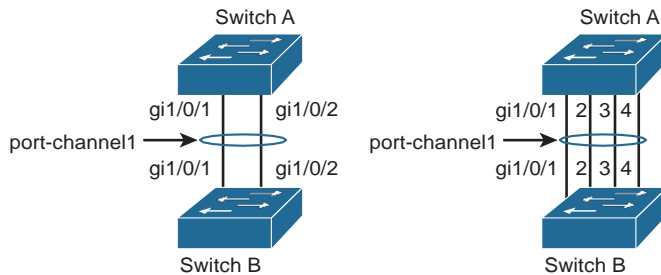


Figure 10-2 *Scaling Bandwidth by Bundling Physical Links into an EtherChannel*

Although an EtherChannel link is seen as a single logical link, the link does not necessarily have an inherent total bandwidth equal to the sum of its component physical links. For example, suppose that a GEC link is made up of four full-duplex 1-Gbps GE links. Although it is possible for the GEC link to carry a total throughput of 8 Gbps (if each link becomes fully loaded), the single resulting GEC bundle does not operate at this speed.

Instead, traffic is distributed across the individual links within the EtherChannel. Each of these links operates at its inherent speed (2 Gbps full duplex for GE) but carries only the frames placed on it by the EtherChannel hardware. If the load-distribution algorithm favors one link within the bundle, that link will carry a disproportionate amount of traffic. In other words, the load is not always distributed equally among the individual links. The load-balancing process is explained further in the next section.

EtherChannel also provides redundancy with several bundled physical links. If one of the links within the bundle fails, traffic sent through that link is automatically moved to an adjacent link. Failover occurs in less than a few milliseconds and is transparent to the end user. As more links fail, more traffic is moved to further adjacent links. Likewise, as links are restored, the load automatically is redistributed among the active links.

As you plan on configuring an EtherChannel, you should give some thought to several different failure scenarios. For example, the failure of a single physical link within an EtherChannel is not catastrophic because the switches compensate by moving traffic to the other, still functioning, links. However, what if all of the physical links are connected to the same switch, as shown in Figure 10-2? The switch itself might fail someday, taking all of the physical links of the EtherChannel down with it.

A more robust solution involves distributing the physical links across multiple switches at each end of the EtherChannel. This is possible when the switches are configured as

one logical or virtual switch, such as the Cisco stackable Catalyst switches or chassis-based Virtual Switching System (VSS) switch families. In Figure 10-3, a four-port GEC is made up of two links connected to the first switch in a stack and two links connected to the second switch in a stack. This is known as a *multichassis EtherChannel* (MEC). Even if one switch fails within a stack, the MEC will keep functioning thanks to the other stacked switch.

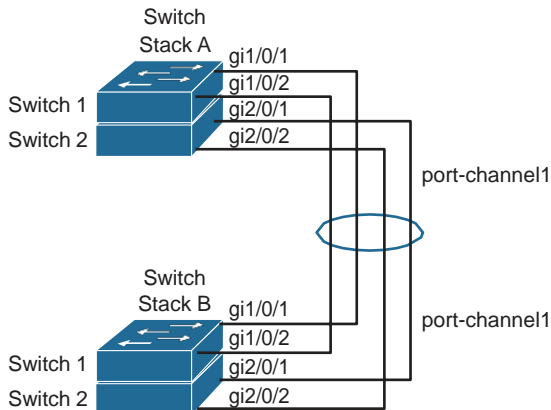


Figure 10-3 Increasing Availability with a Multichassis EtherChannel

Bundling Ports with EtherChannel

EtherChannel bundles can consist of up to eight physical ports of the same Ethernet media type and speed. Some configuration restrictions exist to ensure that only similarly configured links are bundled.

Generally, all bundled ports first must belong to the same VLAN. If used as a trunk, bundled ports must be in trunking mode, have the same native VLAN, and pass the same set of VLANs. Each of the ports should have the same speed and duplex settings before being bundled. Bundled ports also must be configured with identical spanning-tree settings.

Distributing Traffic in EtherChannel



Traffic in an EtherChannel is distributed across the individual bundled links in a deterministic fashion; however, the load is not necessarily balanced equally across all the links. Instead, frames are forwarded on a specific link as a result of a hashing algorithm. The algorithm can use source IP address, destination IP address, or a combination of source and destination IP addresses, source and destination MAC addresses, or TCP/UDP port numbers. The hash algorithm computes a binary pattern that selects a link number in the bundle to carry each frame.

If only one address or port number is hashed, a switch forwards each frame by using one or more low-order bits of the hash value as an index into the bundled links. If two

addresses or port numbers are hashed, a switch performs an exclusive-OR (XOR) operation on one or more low-order bits of the addresses or TCP/UDP port numbers as an index into the bundled links.

For example, an EtherChannel consisting of two links bundled together requires a 1-bit index. If the index is 0, link 0 is selected; if the index is 1, link 1 is used. Either the lowest-order address bit or the XOR of the last bit of the addresses in the frame is used as the index. A four-link bundle uses a hash of the last 2 bits. Likewise, an eight-link bundle uses a hash of the last 3 bits. The hashing operation's outcome selects the EtherChannel's outbound link. Table 10-2 shows the results of an XOR on a two-link bundle, using the source and destination addresses.

Table 10-2 *Frame Distribution on a Two-Link EtherChannel*

Binary Address	Two-Link EtherChannel XOR and LinkNumber
Addr1: ... xxxxxxx0	... xxxxxxx0: Use link 0
Addr2: ... xxxxxxx0	
Addr1: ... xxxxxxx0	... xxxxxxx1: Use link 1
Addr2: ... xxxxxxx1	
Addr1: ... xxxxxxx1	... xxxxxxx1: Use link 1
Addr2: ... xxxxxxx0	
Addr1: ... xxxxxxx1	... xxxxxxx0: Use link 0
Addr2: ... xxxxxxx1	

The XOR operation is performed independently on each bit position in the address value. If the two address values have the same bit value, the XOR result is always 0. If the two address bits differ, the XOR result is always 1. In this way, frames can be distributed statistically among the links with the assumption that MAC or IP addresses themselves are distributed statistically throughout the network. In a four-link EtherChannel, the XOR is performed on the lower 2 bits of the address values, resulting in a 2-bit XOR value (each bit is computed separately) or a link number from 0 to 3.

As an example, consider a packet being sent from IP address 192.168.1.1 to 172.31.67.46. Because EtherChannels can be built from two to eight individual links, only the rightmost (least-significant) 3 bits are needed as a link index. From the source and destination addresses, these bits are 001 (1) and 110 (6), respectively. For a two-link EtherChannel, a 1-bit XOR is performed on the rightmost address bit: 1 XOR 0 = 1, causing Link 1 in the bundle to be used. A four-link EtherChannel produces a 2-bit XOR: 01 XOR 10 = 11, causing Link 3 in the bundle to be used. Finally, an eight-link EtherChannel requires a 3-bit XOR: 001 XOR 110 = 111, where Link 7 in the bundle is selected.

A conversation between two devices always is sent through the same EtherChannel link because the two endpoint addresses stay the same. However, when a device talks to sev-

eral other devices, chances are that the destination addresses are distributed equally with 0s and 1s in the last bit (even and odd address values). This causes the frames to be distributed across the EtherChannel links.

Note that the load distribution is still proportional to the volume of traffic passing between pairs of hosts or link indexes. For example, suppose that there are two pairs of hosts talking across a two-link channel, and each pair of addresses results in a unique link index. Frames from one pair of hosts always travel over one link in the channel, whereas frames from the other pair travel over the other link. The links are both being used as a result of the hash algorithm, so the load is being distributed across every link in the channel.

However, if one pair of hosts has a much greater volume of traffic than the other pair, one link in the channel will be used much more than the other. This still can create a load imbalance. To remedy this condition, you should consider other methods of hashing algorithms for the channel. For example, a method that combines the source and destination addresses along with UDP or TCP port numbers in a single XOR operation can distribute traffic much differently. Then, packets are placed on links within the bundle based on the applications (port numbers) used within conversations between two hosts. The possible hashing methods are discussed in the following section.

Configuring EtherChannel Load Balancing

The hashing operation can be performed on either MAC or IP addresses and can be based solely on source or destination addresses, or both. Use the following command to configure frame distribution for all EtherChannel switch links:

```
Switch(config)# port-channel load-balance method
```

Notice that the load-balancing method is set with a global configuration command. You must set the method globally for the switch, not on a per-port basis. Table 10-3 lists the possible values for the method variable, along with the hashing operation and some sample supporting switch models.

Table 10-3 *Types of EtherChannel Load-Balancing Methods*

Method Value	Hash Input	Hash Operation	Switch Model
src-ip	Source IP address	Bits	All models
dst-ip	Destination IP address	Bits	All models
src-dst-ip	Source and destination IP address	XOR	All models
src-mac	Source MAC address	Bits	All models
dst-mac	Destination MAC address	Bits	All models
src-dst-mac	Source and destination MAC	XOR	All models
src-port	Source port number	Bits	4500, 6500

Method Value	Hash Input	Hash Operation	Switch Model
dst-port	Destination port number	Bits	4500, 6500
src-dst-port	Source and destination port	XOR	4500, 6500

The default configuration depends on the switch model and hardware capabilities. In common access layer switch models such as the Catalyst 3750-X, the default is **src-mac**. You can verify the load-balancing method currently in use with the **show etherchannel load-balance** command, as shown in Example 10-1.

Example 10-1 *Displaying the Current EtherChannel Load-Balancing Method*

```
Switch# show etherchannel load-balance
EtherChannel Load-Balancing Configuration:
    src-mac

EtherChannel Load-Balancing Addresses Used Per-Protocol:
Non-IP: Source MAC address
  IPv4: Source MAC address
  IPv6: Source MAC address
Switch#
```

Normally, the default action should result in a statistical distribution of frames; however, you should determine whether the EtherChannel is imbalanced according to the traffic patterns present. For example, if a single server is receiving most of the traffic on an EtherChannel, the server's address (the destination IP address) always will remain constant in the many conversations. This can cause one link to be overused if the destination IP address is used as a component of a load-balancing method. In the case of a four-link EtherChannel, perhaps two of the four links are overused. Configuring the use of MAC addresses, or only the source IP addresses, might cause the distribution to be more balanced across all the bundled links.

Tip To verify how effectively a configured load-balancing method is performing, you can use the **show etherchannel port-channel** command. Each link in the channel is displayed, along with a hex "Load" value. Although this information is not intuitive, you can use the hex values to get an idea of each link's traffic loads relative to the others.

In some applications, EtherChannel traffic might consist of protocols other than IP. For example, IPX or SNA frames might be switched along with IP. Non-IP protocols need to be distributed according to MAC addresses because IP addresses are not applicable. Here, the switch should be configured to use MAC addresses instead of the IP default.

Tip A special case results when a router is connected to an EtherChannel. Recall that a router always uses its burned-in MAC address in Ethernet frames, even though it is forwarding packets to and from many different IP addresses. In other words, all the traffic sent by the router will use the router’s MAC address as the source. As well, many end stations send frames to their local router address with the router’s MAC address as the destination. This means that the destination MAC address is the same for all frames destined through the router.

Usually, this will not present a problem because the source MAC addresses are all different. When two routers are forwarding frames to each other, however, both source and destination MAC addresses remain constant, and only one link of the EtherChannel is used. If the MAC addresses remain constant, choose IP addresses instead. Beyond that, if most of the traffic is between the same two IP addresses, as in the case of two servers talking, choose IP port numbers to disperse the frames across different links.

You should choose the load-balancing method that provides the greatest distribution or variety when the channel links are indexed. Also consider the type of addressing that is being used on the network. If most of the traffic is IP, it might make sense to load balance according to IP addresses or TCP/UDP port numbers.

But if IP load balancing is being used, what happens to non-IP frames? If a frame cannot meet the load-balancing criteria, the switch automatically falls back to the “next lowest” method. With Ethernet, MAC addresses must always be present, so the switch distributes those frames according to their MAC addresses.

A switch also provides some inherent protection against bridging loops with EtherChannels. When ports are bundled into an EtherChannel, no inbound (received) broadcasts and multicasts are sent back out over any of the remaining ports in the channel. Outbound broadcast and multicast frames are load-balanced like any other: The broadcast or multicast address becomes part of the hashing calculation to choose an outbound channel link.

EtherChannel Negotiation Protocols

EtherChannels can be negotiated between two switches to provide some dynamic link configuration. Two protocols are available to negotiate bundled links in Catalyst switches. The Port Aggregation Protocol (PAgP) is a Cisco proprietary solution, and the Link Aggregation Control Protocol (LACP) is standards based. Table 10-4 summarizes the negotiation protocols and their operation.

Table 10-4 *EtherChannel Negotiation Protocols*

Negotiation	Mode	Negotiation Packets Sent?	Characteristics
PAgP	LACP		
On	On	No	All ports channeling
Auto	Passive	Yes	Waits to channel until asked
Desirable	Active	Yes	Actively asks to form a channel

Port Aggregation Protocol



To provide automatic EtherChannel configuration and negotiation between switches, Cisco developed the Port Aggregation Protocol. PAgP packets are exchanged between switches over EtherChannel-capable ports. Neighbors are identified and port group capabilities are learned and compared with local switch capabilities. Ports that have the same neighbor device ID and port group capability are bundled together as a bidirectional, point-to-point EtherChannel link.

PAgP forms an EtherChannel only on ports that are configured for either identical static VLANs or trunking. PAgP also dynamically modifies parameters of the EtherChannel if one of the bundled ports is modified. For example, if the configured VLAN, speed, or duplex mode of a port in an established bundle is changed, PAgP reconfigures that parameter for all ports in the bundle.

PAgP can be configured in active mode (desirable), in which a switch actively asks a far-end switch to negotiate an EtherChannel, or in passive mode (auto, the default), in which a switch negotiates an EtherChannel only if the far end initiates it.

Link Aggregation Control Protocol



LACP is a standards-based alternative to PAgP, defined in IEEE 802.3ad (also known as *IEEE 802.3 Clause 43, "Link Aggregation"*). LACP packets are exchanged between switches over EtherChannel-capable ports. As with PAgP, neighbors are identified and port group capabilities are learned and compared with local switch capabilities. However, LACP also assigns roles to the EtherChannel's endpoints.

The switch with the lowest *system priority* (a 2-byte priority value followed by a 6-byte switch MAC address) is allowed to make decisions about what ports actively are participating in the EtherChannel at a given time.

Ports are selected and become active according to their *port priority* value (a 2-byte priority followed by a 2-byte port number), where a low value indicates a higher priority. A set of up to 16 potential links can be defined for each EtherChannel. Through LACP, a switch selects up to eight of these having the lowest port priorities as active EtherChannel links at any given time. The other links are placed in a standby state and will be enabled in the EtherChannel if one of the active links goes down.

Like PAgP, LACP can be configured in active mode (active), in which a switch actively asks a far-end switch to negotiate an EtherChannel, or in passive mode (passive), in which a switch negotiates an EtherChannel only if the far end initiates it.

EtherChannel Configuration

For each EtherChannel on a switch, you must choose the EtherChannel negotiation protocol and assign individual switch ports to the EtherChannel. Both PAgP- and LACP-negotiated EtherChannels are described in the following sections. You also can configure an EtherChannel to use the on mode, which unconditionally bundles the links. In this case, neither PAgP nor LACP packets are sent or received.

As ports are configured to be members of an EtherChannel, the switch automatically creates a logical port-channel interface. This interface represents the channel as a whole.

Configuring a PAgP EtherChannel

To configure switch ports for PAgP negotiation (the default), use the following commands:



```
Switch(config)# interface type member/module/number
Switch(config-if)# channel-protocol pagp
Switch(config-if)# channel-group number mode {on | {{auto | desirable}
[non-silent]}}
```

Each interface that will be included in a single EtherChannel bundle must be configured and assigned to the same unique channel group *number* (1 to 64). Channel negotiation must be set to **on** (unconditionally channel, no PAgP negotiation), **auto** (passively listen and wait to be asked), or **desirable** (actively ask).

By default, PAgP operates in silent submode with the **desirable** and **auto** modes, and allows ports to be added to an EtherChannel even if the other end of the link is silent and never transmits PAgP packets. This might seem to go against the idea of PAgP, in which two endpoints are supposed to negotiate a channel. After all, how can two switches negotiate anything if no PAgP packets are received?

The key is in the phrase “if the other end is silent.” The silent submode listens for any PAgP packets from the far end, looking to negotiate a channel. If none is received, silent submode assumes that a channel should be built anyway, so no more PAgP packets are expected from the far end.

This allows a switch to form an EtherChannel with a device such as a file server or a network analyzer that does not participate in PAgP. In the case of a network analyzer connected to the far end, you also might want to see the PAgP packets generated by the switch, as if you were using a normal PAgP EtherChannel.

If you expect a PAgP-capable switch to be on the far end, you should add the **non-silent** keyword to the desirable or auto mode. This requires each port to receive PAgP packets before adding them to a channel. If PAgP is not heard on an active port, the port remains in the up state, but PAgP reports to the Spanning Tree Protocol (STP) that the port is down.

Tip In practice, you might notice a delay from the time the links in a channel group are connected until the time the channel is formed and data can pass over it. You will encounter this if both switches are using the default PAgP auto mode and silent submode. Each interface waits to be asked to form a channel, and each interface waits and listens before accepting silent channel partners. The silent submode amounts to approximately a 15-second delay.

Even if the two interfaces are using PAgP auto mode, the link will still eventually come up, although not as a channel. You might notice that the total delay before data can pass over the link is actually approximately 45 or 50 seconds. The first 15 seconds are the result of PAgP silent mode waiting to hear inbound PAgP messages, and the final 30 seconds are the result of the STP moving through the listening and learning stages.

As an example of PAgP configuration, suppose that you want a switch to use an EtherChannel load-balancing hash of both source and destination port numbers. A Gigabit EtherChannel will be built from interfaces Gigabit Ethernet 1/0/1 through 1/0/4, with the switch actively negotiating a channel. The switch should not wait to listen for silent partners. You can use the following configuration commands to accomplish this:

```
Switch(config)# port-channel load-balance src-dst-port
Switch(config)# interface range gig 1/0/1 - 4
Switch(config-if)# channel-protocol pagp
Switch(config-if)# channel-group 1 mode desirable non-silent
```

Configuring a LACP EtherChannel



To configure switch ports for LACP negotiation, use the following commands:

```
Switch(config)# lacp system-priority priority
Switch(config)# interface type member/module/number
Switch(config-if)# channel-protocol lacp
Switch(config-if)# channel-group number mode {on | passive | active}
Switch(config-if)# lacp port-priority priority
```

First, the switch should have its LACP system priority defined (1 to 65,535; default 32,768). If desired, one switch should be assigned a lower system priority than the other so that it can make decisions about the EtherChannel's makeup. Otherwise, both switches will have the same system priority (32,768), and the one with the lower MACaddress will become the decision maker.

Each interface included in a single EtherChannel bundle must be assigned to the same unique channel group *number* (1 to 64). Channel negotiation must be set to **on** (unconditionally channel, no LACP negotiation), **passive** (passively listen and wait to be asked), or **active** (actively ask).

You can configure more interfaces in the channel group number than are allowed to be active in the channel. This prepares extra standby interfaces to replace failed active ones. Use the **lacp port-priority** command to configure a lower port priority (1 to 65,535; default 32,768) for any interfaces that must be active, and a higher priority for interfaces

that might be held in the standby state. Otherwise, just use the default scenario, in which all ports default to 32,768 and the lower port numbers (in interface number order) are used to select the active ports.

As an example of LACP configuration, suppose that you want to configure a switch to negotiate a Gigabit EtherChannel using interfaces Gigabit Ethernet 1/0/1 through 1/0/4 and 2/0/1 through 2/0/4. Interfaces Gigabit Ethernet 1/0/5 through 1/0/8 and 2/0/5 through 2/0/8 are also available, so these can be used as standby links to replace failed links in the channel. This switch should actively negotiate the channel and should be the decision maker about the channel operation.

You can use the following configuration commands to accomplish this:

```
Switch(config)# lacp system-priority 100
Switch(config)# interface range gig 1/0/1 - 4 , gig 2/0/1 - 4
Switch(config-if)# channel-protocol lacp
Switch(config-if)# channel-group 1 mode active
Switch(config-if)# lacp port-priority 100
Switch(config-if)# exit
Switch(config)# interface range gig 1/0/5 - 8 , gig 2/0/5 - 8
Switch(config-if)# channel-protocol lacp
Switch(config-if)# channel-group 1 mode active
```

Notice that interfaces Gigabit Ethernet 1/0/5-8 and 2/0/5-8 have been left to their default port priorities of 32,768. This is higher than the others, which were configured for 100, so they will be held as standby interfaces.

Avoiding Misconfiguration with EtherChannel Guard

Once you configure a set of physical interfaces on one switch to participate in an EtherChannel, you should configure the corresponding interfaces on the neighboring switch. Your goal should be to keep the EtherChannel configurations as predictable as possible, so that nothing unexpected can happen,

What might happen anyway? Suppose that you configure two interfaces on Switch A to form an unconditional EtherChannel that carries all active VLANs. Your associate configures Switch B for the same set of two interfaces, but manages to plug the cables into the wrong two interfaces. It is entirely possible that a bridging loop might form over the dual links because an EtherChannel has not formed on both ends. STP will not operate consistently on all interfaces because Switch A is expecting a working EtherChannel.

If you decide to use PAGP or LACP to negotiate an EtherChannel, the chances of a misconfiguration are slim. An EtherChannel will not be built if it cannot be negotiated on all member links on the switches at both ends.

To reduce the chances of a misconfigured EtherChannel, Cisco Catalyst switches run the EtherChannel Guard feature by default. You can control the feature with the following global configuration command:

```
Switch(config)# [no] spanning-tree etherchannel guard misconfig
```


Notice that the command is directly related to STP operation over an EtherChannel. If a misconfiguration is detected once the interfaces are enabled, the switch will log the problem and will automatically shut the interfaces down and place them in the errdisable state.

In Example 10-2, two interfaces should have formed an EtherChannel, but a misconfiguration has been detected. Notice that both member interfaces, as well as the port channel 1 EtherChannel interface, have been errdisabled. To see the reason behind this action, you can use the **show interfaces status err-disabled** command.

Example 10-2 Detecting EtherChannel Misconfiguration with EtherChannel Guard

```
Mar 30 05:02:16.073: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
Gil/0/25, putting Gil/0/25 in err-disable state
Mar 30 05:02:16.081: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
Gil/0/26, putting Gil/0/26 in err-disable state
Mar 30 05:02:16.115: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
Pol, putting Gil/0/25 in err-disable state
Mar 30 05:02:16.115: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
Pol, putting Gil/0/26 in err-disable stning-ate
Mar 30 05:02:16.115: %PM-4-ERR_DISABLE: channel-misconfig (STP) error detected on
Pol, putting Pol in err-disable state
Mar 30 05:02:17.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
net1/0/25, changed state to down
Mar 30 05:02:17.096: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
net1/0/26, changed state to down
Mar 30 05:02:17.104: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1,
changed state to down
Switch# show interface status err-disabled
```

Port	Name	Status	Reason	Err-disabled Vlans
Gil/0/25		err-disabled	channel-misconfig (STP)	
Gil/0/26		err-disabled	channel-misconfig (STP)	
Pol		err-disabled	channel-misconfig (STP)	

```
Switch#
```

To recover from a misconfigured EtherChannel, first review the interface configuration on both switches. After you have corrected the problem, you must shut down the port channel interface and reenale it. The member interfaces will follow suit automatically, as shown in Example 10-3.

Example 10-3 Reenabling a Misconfigured EtherChannel

```
Switch(config)# interface port-channel1
Switch(config-if)# shutdown
Switch(config-if)# no shutdown
Switch(config-if)# ^Z
Switch#
Mar 30 05:09:21.518: %SYS-5-CONFIG_I: Configured from console by console
Mar 30 05:09:21.719: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/25, changed state
to up
Mar 30 05:09:21.736: %LINK-3-UPDOWN: Interface GigabitEthernet1/0/26, changed state
to up
```

```

Mar 30 05:09:25.536: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
net1/0/25, changed state to up
Mar 30 05:09:25.544: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEther-
net1/0/26, changed state to up
Mar 30 05:09:26.509: %LINK-3-UPDOWN: Interface Port-channel1, changed state to up
Mar 30 05:09:27.516: %LINEPROTO-5-UPDOWN: Line protocol on Interface Port-channel1,
changed state to up
Switch#

```

Troubleshooting an EtherChannel

If you find that an EtherChannel is having problems, remember that the whole concept is based on consistent configurations on both ends of the channel. Here are some reminders about EtherChannel operation and interaction:



- EtherChannel **on** mode does not send or receive PAgP or LACP packets. Therefore, both ends should be set to **on** mode before the channel can form.
- EtherChannel **desirable** (PAgP) or **active** (LACP) mode attempts to ask the far end to bring up a channel. Therefore, the other end must be set to either **desirable** or **auto** mode.
- EtherChannel **auto** (PAgP) or **passive** (LACP) mode participates in the channel protocol, but only if the far end asks for participation. Therefore, two switches in the **auto** or **passive** mode will not form an EtherChannel.
- PAgP **desirable** and **auto** modes default to the silent submode, in which no PAgP packets are expected from the far end. If ports are set to **non-silent** submode, PAgP packets must be received before a channel will form.

First, verify the EtherChannel state with the **show etherchannel summary** command. Each port in the channel is shown, along with flags indicating the port's state, as shown in Example 10-4.

Example 10-4 show etherchannel summary Command Output

```

Switch# show etherchannel summary
Flags: D - down          P - in port-channel
      I - stand-alone s - suspended
      H - Hot-standby (LACP only)
      R - Layer3          S - Layer2
      u - unsuitable for bundling
      U - in use          f - failed to allocate aggregator
      d - default port

Number of channel-groups in use: 1
Number of aggregators:          1

```

Group	Port-channel	Protocol	Ports			
-----+-----+-----+-----						
1	Po1 (SU)	LACP	Gi1/0/1 (P)	Gi1/0/2 (P)	Gi1/0/3 (D)	Gi1/0/4 (P)
			Gi2/0/1 (P)	Gi2/0/2 (P)	Gi2/0/3 (P)	Gi2/0/4 (P)

The status of the port channel shows the EtherChannel logical interface as a whole. This should show SU (Layer 2 channel, in use) if the channel is operational. You also can examine the status of each port within the channel. Notice that most of the channel ports have flags (P), indicating that they are active in the port-channel. One port shows (D) because it is physically not connected or down. If a port is connected but not bundled in the channel, it will have an independent, or (I), flag. You can verify the channel negotiation mode with the **show etherchannel port** command, as shown in Example 10-5. The local switch interface Gigabit Ethernet 1/0/25 is shown using LACP active mode. Notice that you also verify each end's negotiation mode under the Flags heading—the local switch as A (active mode) and the partner, or far end switch, as P (passive mode).

Example 10-5 show etherchannel port *Command Output*

```
Switch# show etherchannel port

Channel-group listing:
-----

Group: 1
-----

Ports in the group:
-----

Port: Gi1/0/25
-----

Port state      = Up Mstr Assoc In-Bndl
Channel group = 1          Mode = Active          Gcchange = -
Port-channel = Po1        GC      = -            Pseudo port-channel = Po1
Port index      = 0        Load = 0x00          Protocol = LACP

Flags:  S - Device is sending Slow LACPDUS      F - Device is sending fast LACPDUS.
        A - Device is in active mode.            P - Device is in passive mode.

Local information:

Port      Flags  State  LACP port  Admin  Oper  Port      Port
Port      Flags  State  Priority   Key    Key   Number    State
Gi1/0/25 SA    bndl    32768     0x1    0x1    0x11A     0x3D

Partner's information:

LACP port
Admin Oper  Port  Port
key  Key   Number State
Port  Flags  Priority Dev ID      Age
```

```
Gil/0/25 SP 32768          aca0.164e.8280  21s    0x0    0x1    0x21A  0x3C

Age of the port in the current state: 0d:00h:02m:37s
[output truncated for clarity]
```

Within a switch, an EtherChannel cannot form unless each of the component or member ports is configured consistently. Each must have the same switch mode (access or trunk), native VLAN, trunked VLANs, port speed, port duplex mode, and so on.

You can display a port’s configuration by looking at the **show running-config interface type mod/ num** output. Also, the **show interface type member/module/number ether-channel** shows all active EtherChannel parameters for a single port. If you configure a port inconsistently with others for an EtherChannel, you see error messages from the switch.

Some messages from the switch might look like errors but are part of the normal EtherChannel process. For example, as a new port is configured as a member of an existing EtherChannel, you might see this message:

```
4d00h: %EC-5-L3DONTBNDL2: GigabitEthernet1/0/11 suspended:
incompatible partner port with GigabitEthernet1/0/10
```

When the port is first added to the EtherChannel, it is incompatible because the STP runs on the channel and the new port. After STP takes the new port through its progression of states, the port is automatically added into the EtherChannel.

Other messages do indicate a port-compatibility error. In these cases, the cause of the error is shown. For example, the following message announces that Gigabit Ethernet1/0/3 has a different duplex mode than the other ports in the EtherChannel:

```
4d00h: %EC-5-CANNOT_BUNDLE2: GigabitEthernet1/0/3 is not compatible
with GigabitEthernet1/0/1 and will be suspended (duplex of Gil/0/3
is full, Gil/0/1 is half)
```

Finally, you can verify the EtherChannel load-balancing or hashing algorithm with the **show etherchannel load-balance** command. Remember that the switches on either end of an EtherChannel can have different load-balancing methods. The only drawback to this is that the load balancing will be asymmetric in the two directions across the channel.

Table 10-5 lists the commands useful for verifying or troubleshooting EtherChannel operation.

Table 10-5 *EtherChannel Troubleshooting Commands*

Display Function	Command Syntax
Current EtherChannel status of each member port	show etherchannel summary
	show etherchannel port
Time stamps of EtherChannel changes	show etherchannel port-channel

Display Function	Command Syntax
Detailed status about each EtherChannel component	show etherchannel detail
Load-balancing hashing algorithm	show etherchannel load-balance
Load-balancing port index used by hashing algorithm	show etherchannel port-channel
EtherChannel neighbors oneach port	show {pagp lacp} neighbor
LACP system ID	show lacp sys-id

Foundation Topics

Inter-VLAN Routing

Recall that a Layer 2 network is defined as a broadcast domain. A Layer 2 network can also exist as a VLAN inside one or more switches. VLANs essentially are isolated from each other so that packets in one VLAN cannot cross into another VLAN.

To transport packets between VLANs, you must use a Layer 3 device. Traditionally, this has been a router's function. The router must have a physical or logical connection to each VLAN so that it can forward packets between them. This is known as *inter-VLAN routing*.



Inter-VLAN routing can be performed by an external router that connects to each of the VLANs on a switch. Separate physical connections can be used, or the router can access each of the VLANs through a single trunk link. Part A of Figure 11-1 illustrates this concept. The external router also can connect to the switch through a single trunk link, carrying all the necessary VLANs, as illustrated in Part B of Figure 11-1. Part B illustrates what commonly is referred to as a “router-on-a-stick” or a “one-armed router” because the router needs only a single interface to do its job.

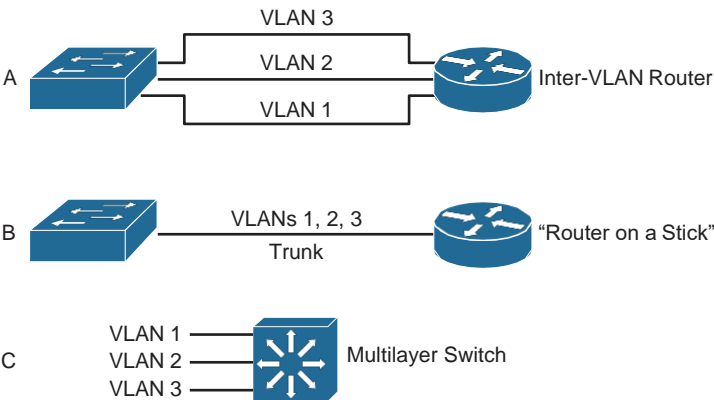


Figure 11-1 Examples of Inter-VLAN Routing Connections

Finally, Part C of Figure 11-1 shows how the routing and switching functions can be combined into one device: a multilayer switch. No external router is needed.

Types of Interfaces

Multilayer switches can perform both Layer 2 switching and inter-VLAN routing, as appropriate. Layer 2 switching occurs between interfaces that are assigned to Layer 2 VLANs or Layer 2 trunks. Layer 3 switching can occur between any type of interface, as long as the interface can have a Layer 3 address assigned to it.



As with a router, a multilayer switch can assign a Layer 3 address to a physical interface. It also can assign a Layer 3 address to a logical interface that represents an entire VLAN. This is known as a *switched virtual interface* (SVI), sometimes called a *switch virtual interface* (SVI). Keep in mind that the Layer 3 address you configure becomes the default gateway for any hosts that are connected to the interface or VLAN. The hosts will use the Layer 3 interface to communicate outside of their local broadcast domains.

Configuring Inter-VLAN Routing

Inter-VLAN routing first requires that routing be enabled for the Layer 3 protocol. In the case of IP, you would enable IP routing. In addition, you must configure static routes or a dynamic routing protocol. These topics are covered fully in the CCNP ROUTE course. By default, every switch port on most Catalyst switch platforms is a Layer 2 interface, whereas every switch port on a Catalyst 6500 is a Layer 3 interface. If an interface needs to operate in a different mode, you must explicitly configure it.

An interface is either in Layer 2 or Layer 3 mode, depending on the use of the **switchport** interface configuration command. You can display a port's current mode with the following command:

```
Switch# show interface type member/module/number switchport
```

If the **switchport:** line in the command output is shown as enabled, the port is in Layer 2 mode. If this line is shown as disabled, as in the following example, the port is in Layer 3 mode:

```
Switch# show interface gigabitethernet 1/0/1 switchport
Name: Gi1/0/1
Switchport: Disabled
Switch#
```

Tip Whenever you see the term *switch port*, think Layer 2. So if the switch port is disabled, it must be Layer 3.

Figure 11-2 shows how the different types of interface modes can be used within a single switch.

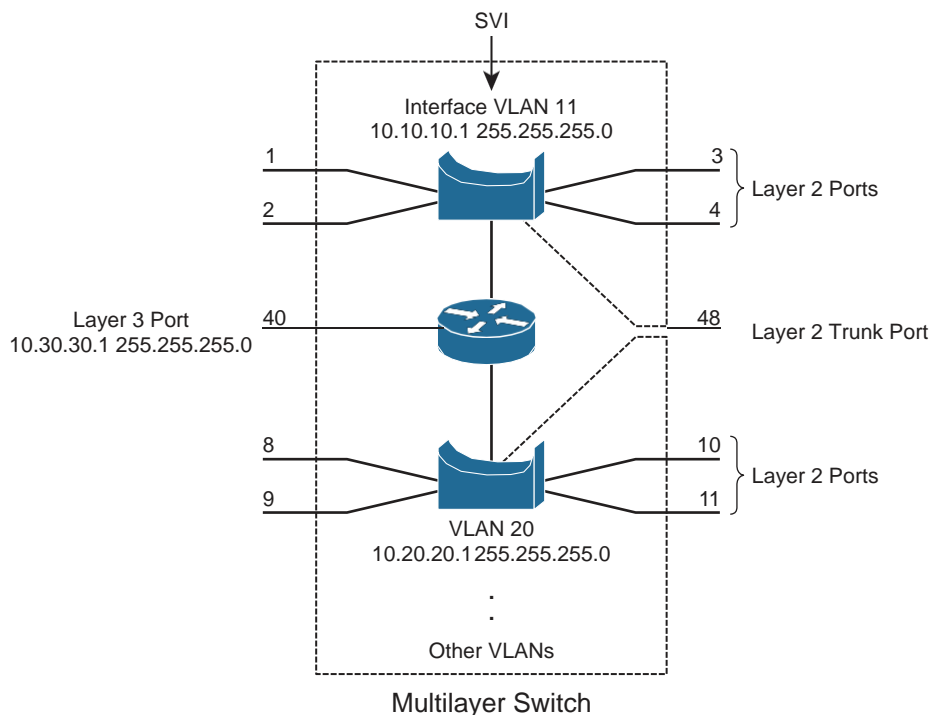


Figure 11-2 Catalyst Switch with Various Types of Ports

Layer 2 Port Configuration



If an interface is in Layer 3 mode and you need to reconfigure it for Layer 2 functionality instead, use the following command sequence:

```
Switch(config)# interface type member/module/number
Switch(config-if)# switchport
```

The **switchport** command puts the port in Layer 2 mode. Then you can use other **switchport** command keywords to configure trunking, access VLANs, and so on. As displayed in Figure 11-2, several Layer 2 ports exist, each assigned to a specific VLAN. A Layer 2 port also can act as a trunk, transporting multiple Layer 2 VLANs.

Layer 3 Port Configuration



Physical switch ports also can operate as Layer 3 interfaces, where a Layer 3 network address is assigned and routing can occur, as shown previously in Figure 11-2. For Layer 3 functionality, you must explicitly configure switch ports with the following command sequence:

```
Switch(config)# interface type member/module/number
Switch(config-if)# no switchport
Switch(config-if)# ip address ip-address mask [secondary]
```

The **no switchport** command takes the port out of Layer 2 operation. You then can assign a network address to the port, as you would to a router interface.

Tip By default, a Catalyst switch sets aside the appropriate amounts of TCAM space to perform Layer 3 operation for IPv4. If you intend to use IPv6 also, be sure to reconfigure the SDM template with the **sdm prefer dual-ipv4-and-ipv6** command.

Note Keep in mind that a Layer 3 port assigns a network address to one specific physical interface. If several interfaces are bundled as an EtherChannel, the EtherChannel can also become a Layer 3 port. In that case, the network address is assigned to the port-channel interface—not to the individual physical links within the channel.

SVI Port Configuration

On a multilayer switch, you also can enable Layer 3 functionality for an entire VLAN on the switch. This allows a network address to be assigned to a logical interface—that of the VLAN itself. This is useful when the switch has many ports assigned to a common VLAN, and routing is needed in and out of that VLAN.

In Figure 11-2, you can see how an IP address is applied to the SVI called VLAN 10. Notice that the SVI itself has no physical connection to the outside world; to reach the outside, VLAN 10 must extend through a Layer 2 port or trunk to the outside.



The logical Layer 3 interface is known as an *SVI*. However, when it is configured, it uses the much more intuitive interface name **vlan *vlan-id***, as if the VLAN itself is a physical interface. First, define or identify the VLAN interface; then assign any Layer 3 functionality to it with the following configuration commands:

```
Switch(config)# interface vlan vlan-id
Switch(config-if)# ip address ip-address mask [secondary]
```

The VLAN must be defined and active on the switch before the SVI can be used. Make sure that the new VLAN interface also is enabled with the **no shutdown** interface configuration command.

Note The VLAN and the SVI are configured separately, even though they interoperate. Creating or configuring the SVI does not create or configure the VLAN; you still must define each one independently.

As an example, the following commands show how VLAN 100 is created and then defined as a Layer 3 SVI:

```
Switch(config)# vlan 100
Switch(config-vlan)# name Example_VLAN
Switch(config-vlan)# exit
Switch(config)# interface vlan 100
Switch(config-if)# ip address 192.168.100.1 255.255.255.0
Switch(config-if)# no shutdown
```

Be aware that an SVI cannot become active until at least one Layer 2 port assigned to the VLAN has also become active and STP has converged. By automatically keeping the SVI down until the VLAN is ready, no other switching or routing functions can attempt to use the SVI prematurely. This function is called SVI autostate.

You might sometimes want the SVI to stay up even when no Layer 2 ports are active on the VLAN. For example, you might have a Layer 2 port configured for port mirroring to capture traffic. In that case, the port would not be up and functioning normally, so it should be excluded from affecting the state of the SVI. You can exclude a switch port with the following interface configuration command:

```
Switch(config-if)# switchport autostate exclude
```

Multilayer Switching with CEF

Catalyst switches can use several methods to forward packets based on Layer 3 and Layer 4 information. The current generation of Catalyst multilayer switches uses the efficient Cisco Express Forwarding (CEF) method. This section describes the evolution of multilayer switching and discusses CEF in detail. Although CEF is easy to configure and use, the underlying switching mechanisms are more involved and should be understood.

Traditional MLS Overview

Multilayer switching began as a dual effort between a route processor (RP) and a switching engine (SE). The basic idea is to “route once and switch many.” The RP receives the first packet of a new traffic flow between two hosts, as usual. A routing decision is made, and the packet is forwarded toward the destination.

To participate in multilayer switching, the SE must know the identity of each RP. The SE then can listen in to the first packet going to the router and also going away from the router. If the SE can switch the packet in both directions, it can learn a “shortcut path” so that subsequent packets of the same flow can be switched directly to the destination port without passing through the RP.

This technique also is known as *NetFlow switching* or *route cache switching*.

Traditionally, NetFlow switching was performed on legacy Cisco hardware, such as the Catalyst 6000 Supervisor 1/1a and Multilayer Switch Feature Card (MSFC), Catalyst 5500 with a Route Switch Module (RSM), Route Switch Feature Card (RSFC), or external router. Basically, the hardware consisted of an independent RP component and a NetFlow-capable SE component.

CEF Overview

NetFlow switching has given way to a more efficient form of multilayer switching: Cisco Express Forwarding. Cisco developed CEF for its line of routers, offering high-performance packet forwarding through the use of dynamic lookup tables. CEF also has been carried over to the Catalyst switching platforms. CEF runs by default, taking advantage of the specialized hardware.

A CEF-based multilayer switch consists of two basic functional blocks, as shown in Figure 11-3: The Layer 3 engine is involved in building routing information that the Layer 3 forwarding engine can use to switch packets in hardware.

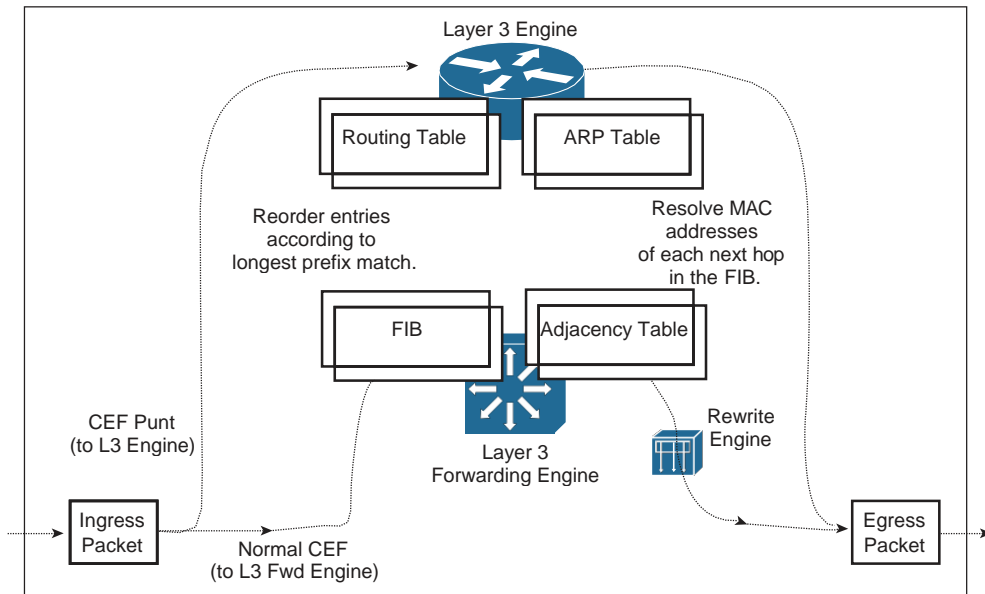


Figure 11-3 Packet Flow Through a CEF-Based Multilayer Switch

Forwarding Information Base



The Layer 3 engine (essentially a router) maintains routing information, whether from static routes or dynamic routing protocols. Basically, the routing table is reformatted into an ordered list with the most specific route first, for each IP destination subnet in the table. The new format is called a Forwarding Information Base (FIB) and contains routing or forwarding information that the network prefix can reference.

In other words, a route to 10.1.0.0/16 might be contained in the FIB along with routes to 10.1.1.0/24 and 10.1.1.128/25, if those exist. Notice that these examples are increasingly more specific subnets, as designated by the longer subnet masks. In the FIB, these would be ordered with the most specific, or longest match, first, followed by less specific subnets. When the switch receives a packet, it easily can examine the destination address and find the longest-match destination route entry in the FIB.

The FIB also contains the next-hop address for each entry. When a longest-match entry is found in the FIB, the Layer 3 next-hop address is found, too.

You might be surprised to know that the FIB also contains host route (subnet mask 255.255.255.255) entries. These normally are not found in the routing table unless they are advertised or manually configured. Host routes are maintained in the FIB for the most efficient routing lookup to directly connected or adjacent hosts.

As with a routing table, the FIB is dynamic in nature. When the Layer 3 engine sees a change in the routing topology, it sends an update to the FIB. Anytime the routing table receives a change to a route prefix or the next-hop address, the FIB receives the same change. Also, if a next-hop address is changed or aged out of the Address Resolution Protocol (ARP) table, the FIB must reflect the same change.

You can display FIB table entries related to a specific interface or VLAN with the following form of the **show ip cef** command:

```
Switch# show ip cef [type member/module/number | vlan vlan-id] [detail]
```

The FIB entries corresponding to the VLAN 101 switched virtual interface might be shown as demonstrated in Example 11-1.

Example 11-1 *Displaying FIB Table Entries for a Specified VLAN*

```
Switch# show ip cef vlan 101
Prefix          Next Hop      Interface
10.1.1.0/24     attached     Vlan101
10.1.1.2/32     10.1.1.2    Vlan101
10.1.1.3/32     10.1.1.3    Vlan101
Switch#
```

You also can view FIB entries by specifying an IP prefix address and mask, using the following form of the **show ip cef** command:

```
Switch# show ip cef [prefix-ip prefix-mask] [longer-prefixes] [detail]
```

The output in Example 11-2 displays any subnet within 10.1.0.0/16 that is known by the switch, regardless of the prefix or mask length. Normally, only an exact match of the IP prefix and mask will be displayed if it exists in the CEF table. To see other longer match entries, you can add the **longer-prefixes** keyword.

Example 11-2 *Displaying FIB Table Entries for a Specified IP Prefix Address/Mask*

```
Switch# show ip cef 10.1.0.0 255.255.0.0 longer-prefixes
Prefix          Next Hop      Interface
10.1.1.0/24     attached     Vlan101
10.1.1.2/32     10.1.1.2    Vlan101
10.1.1.3/32     10.1.1.3    Vlan101
10.1.2.0/24     attached     Vlan102
10.1.3.0/26     192.168.1.2  Vlan99
                  192.168.1.3  Vlan99
10.1.3.64/26    192.168.1.2  Vlan99
                  192.168.1.3  Vlan99
10.1.3.128/26   192.168.1.4  Vlan99
                  192.168.1.3  Vlan99

[output omitted]
Switch#
```

Notice that the first three entries are the same ones listed in Example 11-1. Other subnets also are displayed, along with their next-hop router addresses and switch interfaces.

You can add the **detail** keyword to see more information about each FIB table entry for CEF, as demonstrated in Example 11-3.

Example 11-3 *Displaying Detailed CEF Entry Information*

```
Switch# show ip cef 10.1.3.0 255.255.255.192 detail
10.1.3.0/26, version 270, epoch 0, per-destination sharing
0 packets, 0 bytes
  via 192.168.1.2, Vlan99, 0 dependencies
    traffic share 1
    next hop 192.168.1.2, Vlan99
    valid adjacency
  via 192.168.1.3, Vlan99, 0 dependencies
    traffic share 1
    next hop 192.168.1.3, Vlan99
    valid adjacency
0 packets, 0 bytes switched through the prefix
tmstats: external 0 packets, 0 bytes
         internal 0 packets, 0 bytes
Switch#
```

The version number describes the number of times the CEF entry has been updated since the table was generated. The epoch number denotes the number of times the CEF table has been flushed and regenerated as a whole. The 10.1.3.0/26 subnet has two next-hop router addresses, so the local switch is using per-destination load sharing between the two routers.

After the FIB is built, packets can be forwarded along the bottom dashed path in Figure 11-3. This follows the hardware switching process, in which no “expensive” or time-consuming operations are needed. At times, however, a packet cannot be switched in hardware, according to the FIB. Packets then are marked as “CEF punt” and immediately are sent to the Layer 3 engine for further processing, as shown in the top dashed path in Figure 11-3. Some of the conditions that can cause this are as follows:

- An entry cannot be located in the FIB.
- The FIB table is full.
- The IP Time-To-Live (TTL) has expired.
- The maximum transmission unit (MTU) is exceeded, and the packet must be fragmented.
- An Internet Control Message Protocol (ICMP) redirect is involved.
- The encapsulation type is not supported.
- Packets are tunneled, requiring a compression or encryption operation.

- An access list with the **log** option is triggered.
- A Network Address Translation (NAT) operation must be performed.

CEF operations can be handled on a single, fixed hardware platform. The FIB is generated and contained centrally in the switch. CEF also can be optimized through the use of specialized forwarding hardware, using the following techniques:

- **Accelerated CEF (aCEF):** CEF is distributed across multiple Layer 3 forwarding engines, typically located on individual line cards in chassis-based Catalyst switches. These engines do not have the capability to store and use the entire FIB, so only a portion of the FIB is downloaded to them at any time. This functions as an FIB “cache,” containing entries that are likely to be used again. If FIB entries are not found in the cache, requests are sent to the Layer 3 engine for more FIB information. The net result is that CEF is accelerated on the line cards, but not necessarily at a sustained wire-speed rate.
- **Distributed CEF (dCEF):** CEF can be distributed completely among multiple Layer 3 forwarding engines for even greater performance. Because the FIB is self-contained for complete Layer 3 forwarding, it can be replicated across any number of independent Layer 3 forwarding engines. For example, the Catalyst 6500 has line cards that support dCEF, each with its own FIB table and forwarding engine. A central Layer 3 engine maintains the routing table and generates the FIB, which is then dynamically downloaded in full to each of the line cards.

Adjacency Table

A router normally maintains a routing table containing Layer 3 network and next-hop information, and an ARP table containing Layer 3 to Layer 2 address mapping. These tables are kept independently.



Recall that the FIB keeps the Layer 3 next-hop address for each entry. To streamline packet forwarding even more, the FIB has corresponding Layer 2 information for every next-hop entry. This portion of the FIB is called the *adjacency table*, consisting of the MAC addresses of nodes that can be reached in a single Layer 2 hop.

You can display the adjacency table’s contents with the following command:

```
Switch# show adjacency [type member/module/number | vlan vlan-id] [summary | detail]
```

As an example, the total number of adjacencies known on each physical or VLAN interface can be displayed with the **show adjacency summary** command, as demonstrated in Example 11-4.

Example 11-4 *Displaying the Total Number of Known Adjacencies*

```
Switch# show adjacency summary
Adjacency Table has 106 adjacencies
  Table epoch: 0 (106 entries at this epoch)
  Interface          Adjacency Count
  Vlan99              21
  Vlan101              3
  Vlan102              1
  Vlan103              47
  Vlan104              7
  Vlan105              27
Switch#
```

Adjacencies are kept for each next-hop router and each host that is connected directly to the local switch. You can see more detailed information about the adjacencies by using the **detail** keyword, as demonstrated in Example 11-5.

Example 11-5 *Displaying Detailed Information About Adjacencies*

```
Switch# show adjacency vlan 99 detail
Protocol Interface          Address
IP          Vlan99          192.168.1.2(5)
                                0 packets, 0 bytes
                                epoch 0
                                sourced in sev-epoch 0
                                Encap length 14
                                000A5E45B145000E387D51000800
                                L2 destination address byte offset 0
                                L2 destination address byte length 6
                                Link-type after encap: ip
                                ARP
IP          Vlan99          192.168.1.3(5)
                                1 packets, 104 bytes
                                L2 destination address byte offset 0
                                L2 destination address byte length 6
                                Link-type after encap: ip
                                ARP
                                000CF1C909A0000E387D51000800
                                L2 destination address byte offset 0
                                L2 destination address byte length 6
                                Link-type after encap: ip
                                ARP
```


Notice that the adjacency entries include both the IP address (Layer 3) and the MAC address (Layer 2) of the directly attached host. The MAC address could be shown as the first six octets of the long string of hex digits (as shaded in the previous output) or on a line by itself. The remainder of the string of hex digits contains the MAC address of the Layer 3 engine's interface (six octets, corresponding to the Vlan99 interface in the example) and the EtherType value (two octets, where 0800 denotes IP).

The adjacency table information is built from the ARP table. Example 11-5 shows adjacency with the age of its ARP entry. As a next-hop address receives a valid ARP entry, the adjacency table is updated. If an ARP entry does not exist, the FIB entry is marked as "CEF glean." This means that the Layer 3 forwarding engine cannot forward the packet in hardware because of the missing Layer 2 next-hop address. The packet is sent to the Layer 3 engine so that it can generate an ARP request and receive an ARP reply. This is known as the *CEF glean state*, in which the Layer 3 engine must glean the next-hop destination's MAC address.

The glean state can be demonstrated in several ways, as demonstrated in Example 11-6.

Example 11-6 Displaying Adjacencies in the CEF Glean State

```
Switch# show ip cef adjacency glean
Prefix                Next Hop                Interface
10.1.1.2/32           attached                Vlan101
127.0.0.0/8           attached                EOBC0/0
[output omitted]
Switch# show ip arp 10.1.1.2
Switch# show ip cef 10.1.1.2 255.255.255.255 detail
10.1.1.2/32, version 688, epoch 0, attached, connected
0 packets, 0 bytes
  via Vlan101, 0 dependencies
    valid glean adjacency
Switch#
```

Notice that the FIB entry for directly connected host 10.1.1.2/32 is present but listed in the glean state. The **show ip arp** command shows that there is no valid ARP entry for the IP address.

During the time that an FIB entry is in the CEF glean state waiting for the ARP resolution, subsequent packets to that host are immediately dropped so that the input queues do not fill and the Layer 3 engine does not become too busy worrying about the need for duplicate ARP requests. This is called *ARP throttling* or *throttling adjacency*. If an ARP reply is not received in 2 seconds, the throttling is released so that another ARP request can be triggered. Otherwise, after an ARP reply is received, the throttling is released, the FIB entry can be completed, and packets can be forwarded completely in hardware.

The adjacency table also can contain other types of entries so that packets can be handled efficiently. For example, you might see the following adjacency types listed:

- **Null adjacency:** Used to switch packets destined for the null interface. The null interface always is defined on a router or switch; it represents a logical interface that silently absorbs packets without actually forwarding them.

- **Drop adjacency:** Used to switch packets that cannot be forwarded normally. In effect, these packets are dropped without being forwarded. Packets can be dropped because of an encapsulation failure, an unresolved address, an unsupported protocol, no valid route present, no valid adjacency, or a checksum error. You can gauge drop adjacency activity with the following command:

```
Switch# show cef drop
CEF Drop Statistics
Slot  Encap_fail  Unresolved  Unsupported  No_route  No_adj  ChkSum_Err
RP      8799327      1          45827      5089667    32      0
Switch#
```

- **Discard adjacency:** Used when packets must be discarded because of an access list or other policy action.
- **Punt adjacency:** Used when packets must be sent to the Layer 3 engine for further processing. You can gauge the CEF punt activity by looking at the various punt adjacency reasons listed by the **show cef not-cef-switched** command:

```
Switch# show cef not-cef-switched
CEF Packets passed on to next switching layer
Slot No_adj No_encap Unsupp'ted Redirect Receive Options Access Frag
RP   3579706      0      0      0 41258564      0      0      0
Switch#
```

The reasons shown are as follows:

- **No_adj:** An incomplete adjacency
- **No_encap:** An incomplete ARP resolution
- **Unsupp'ted:** Unsupported packet features
- **Redirect:** ICMP redirect
- **Receive:** Layer 3 engine interfaces; includes packets destined for IP addresses that are assigned to interfaces on the Layer 3 engine, IP network addresses, and IP broadcast addresses
- **Options:** IP options present
- **Access:** Access list evaluation failure
- **Frag:** Fragmentation failure

Packet Rewrite

When a multilayer switch finds valid entries in the FIB and adjacency tables, a packet is almost ready to be forwarded. One step remains: The packet header information must be rewritten. Keep in mind that multilayer switching occurs as quick table lookups to find the next-hop address and the outbound switch port. The packet is untouched and still has the original destination MAC address of the switch itself. The IP header also must be adjusted, as if a traditional router had done the forwarding.

The switch has an additional functional block that performs a packet rewrite in real time. The packet rewrite engine (shown in Figure 11-3) makes the following changes to the packet just before forwarding:



- **Layer 2 destination address:** Changed to the next-hop device's MAC address
- **Layer 2 source address:** Changed to the outbound Layer 3 switch interface's MAC address
- **Layer 3 IP TTL:** Decrement by one because one router hop has just occurred
- **Layer 3 IP checksum:** Recalculated to include changes to the IP header
- **Layer 2 frame checksum:** Recalculated to include changes to the Layer 2 and Layer 3 headers

A traditional router normally would make the same changes to each packet. The multilayer switch must act as if a traditional router were being used, making identical changes. However, the multilayer switch can do this very efficiently with dedicated packet-rewrite hardware and address information obtained from table lookups.

Configuring CEF

CEF is enabled on all CEF-capable Catalyst switches by default. In fact, many switches run CEF inherently, so CEF never can be disabled.

Tip Switches such as the Catalyst 3750 and 4500 run CEF by default, but you can disable CEF on a per-interface basis. You can use the **no ip route-cache cef** and **no ip cef** interface configuration commands to disable CEF on the Catalyst 3750 and 4500, respectively.

You should always keep CEF enabled whenever possible, except when you need to disable it for debugging purposes.

Verifying Multilayer Switching

The multilayer switching topics presented in this chapter are not difficult to configure; however, you might need to verify how a switch is forwarding packets. In particular, the following sections discuss the commands that you can use to verify the operation of inter-VLAN routing and CEF.

Verifying Inter-VLAN Routing

To verify the configuration of a Layer 2 port, you can use the following EXEC command:

```
Switch# show interface type member/module/number switchport
```

The output from this command displays the access VLAN or the trunking mode and native VLAN. The administrative modes reflect what has been configured for the port, whereas the operational modes show the port's active status.

You can use this same command to verify the configuration of a Layer 3 or routed port. In this case, you should see the switchport (Layer 2) mode disabled, as in Example 11-7.

Example 11-7 Verifying Configuration of a Layer 3 Switch Port

```
Switch# show interface gigabitethernet 1/0/1 switchport
Name: Gi1/0/1
Switchport: Disabled
Switch#
```

To verify the configuration of an SVI, you can use the following EXEC command:

```
Switch# show interface vlan vlan-id
```

The VLAN interface should be up, with the line protocol also up. If this is not true, either the interface is disabled with the **shutdown** command, the VLAN itself has not been defined on the switch, or there are no active Layer 2 switch interfaces configured to use the VLAN. Use the **show vlan** command to see a list of configured VLANs.

Example 11-8 shows the output produced from the **show vlan** command. Notice that each defined VLAN is shown, along with the switch ports that are assigned to it.

Example 11-8 Displaying a List of Configured VLANs

```
Switch# show vlan
```

VLAN	Name	Status	Ports
1	default	active	Gi1/0/1, Gi1/0/2, Gi1/0/3 Gi1/0/4, Gi1/0/5, Gi1/0/6 Gi1/0/7, Gi1/0/8, Gi1/0/9 Gi1/0/10, Gi1/0/11, Gi1/0/12 Gi1/0/13, Gi1/0/14, Gi1/0/15 Gi1/0/16, Gi1/0/17, Gi1/0/18 Gi1/0/19, Gi1/0/20, Gi1/0/21 Gi1/0/25, Gi1/0/26, Te1/0/1 Te1/0/2
2	VLAN0002	active	Gi1/0/22
5	VLAN0005	active	
10	VLAN0010	active	
11	VLAN0011	active	Gi1/0/23
12	VLAN0012	active	
99	VLAN0099	active	Gi1/0/24

```
Switch#
```

You also can display the IP-related information about a switch interface with the **show ip interface** command, as demonstrated in Example 11-9.

Example 11-9 *Displaying IP-Related Information About a Switch Interface*

```

Switch# show ip interface vlan 101
Vlan101 is up, line protocol is up
  Internet address is 10.1.1.1/24
  Broadcast address is 255.255.255.255
  Address determined by setup command
  MTU is 1500 bytes
  Helper address is not set
  Directed broadcast forwarding is disabled
  Outgoing access list is not set
  Inbound access list is not set
  Proxy ARP is enabled
  Local Proxy ARP is disabled
  Security level is default
  Split horizon is enabled
  ICMP redirects are always sent
  ICMP unreachable are always sent
  ICMP mask replies are never sent
  IP fast switching is enabled
  IP fast switching on the same interface is disabled
  IP Flow switching is disabled
  IP CEF switching is enabled
  IP Feature Fast switching turbo vector
  IP Feature CEF switching turbo vector
  IP multicast fast switching is enabled
  IP multicast distributed fast switching is disabled
  IP route-cache flags are Fast, Distributed, CEF
  Router Discovery is disabled
  IP output packet accounting is disabled
  IP access violation accounting is disabled
  TCP/IP header compression is disabled
  RTP/IP header compression is disabled
  Probe proxy name replies are disabled
  Policy routing is disabled
  Network address translation is disabled
  WCCP Redirect outbound is disabled
  WCCP Redirect inbound is disabled
  WCCP Redirect exclude is disabled
  BGP Policy Mapping is disabled
  Sampled Netflow is disabled
  IP multicast multilayer switching is disabled
Switch#

```

You can use the **show ip interface brief** command to see a summary listing of the Layer 3 interfaces involved in routing IP traffic, as demonstrated in Example 11-10.

Example 11-10 *Displaying a Summary Listing of Interfaces Routing IP Traffic*

```
Switch# show ip interface brief
Interface                IP-Address      OK? Method Status      Protocol
Vlan1                    unassigned      YES NVRAM administratively down  down
Vlan54                   10.3.1.6        YES manual up          up
Vlan101                  10.1.1.1        YES manual up          up
GigabitEthernet1/0/10    10.1.5.1        YES manual up          up
[output omitted]
Switch#
```

Verifying CEF

CEF operation depends on the correct routing information being generated and downloaded to the Layer 3 forwarding engine hardware. This information is contained in the FIB and is maintained dynamically. To view the entire FIB, use the following EXEC command:

```
Switch# show ip cef
```

Example 11-11 shows sample output from this command.

Example 11-11 *Displaying the FIB Contents for a Switch*

```
Switch# show ip cef
Prefix                Next Hop          Interface
0.0.0.0/32            receive
192.168.199.0/24      attached          Vlan1
192.168.199.0/32      receive
192.168.199.1/32      receive
192.168.199.2/32      192.168.199.2    Vlan1
192.168.199.255/32    receive
Switch#
```

On this switch, only VLAN 1 has been configured with the IP address 192.168.199.1 255.255.255.0. Notice several things about the FIB for such a small configuration:

- **0.0.0.0/32:** An FIB entry has been reserved for the default route. No next hop is defined, so the entry is marked “receive” so that packets will be sent to the Layer 3 engine for further processing.
- **192.168.199.0/24:** The subnet assigned to the VLAN 1 interface is given its own entry. This is marked “attached” because it is connected directly to an SVI, VLAN 1.
- **192.168.199.0/32:** An FIB entry has been reserved for the exact network address. This is used to contain an adjacency for packets sent to the network address, if the network is not directly connected. In this case, there is no adjacency, and the entry is marked “receive.”

- **192.168.199.1/32:** An entry has been reserved for the VLAN 1 SVI's IP address. Notice that this is a host route (/32). Packets destined for the VLAN 1 interface must be dealt with internally, so the entry is marked "receive."
- **192.168.199.2/32:** This is an entry for a neighboring multilayer switch, found on the VLAN 1 interface. The Next Hop field has been filled in with the same IP address, denoting that an adjacency is available.
- **192.168.199.255/32:** An FIB entry has been reserved for the 192.168.199.0 subnet's broadcast address. The route processor (Layer 3 engine) handles all directed broadcasts, so the entry is marked "receive."

To see complete FIB table information for a specific interface, use the following EXEC command:

```
Switch# show ip cef type member/module/number [detail]
```

Exam Preparation Tasks

Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the outer margin of the page. Table 11-2 lists a reference of these key topics and the page numbers on which each is found.



Table 11-2 *Key Topics for Chapter 11*

Key Topic Element	Description	Page Number
Paragraph	Describes inter-VLAN routing	268
Paragraph	Describes SVIs	269
Paragraph	Explains Layer 2 interface mode configuration	270
Paragraph	Explains Layer 3 interface mode configuration	270
Paragraph	Explains how to configure an SVI	271
Paragraph	Discusses the FIB and its contents	273
Paragraph	Explains the CEF adjacency table	276
List	Explains which IP packet fields are changed during the packet rewrite process	280

Complete Tables and Lists from Memory

There are no memory tables in this chapter.

Define Key Terms

Define the following key terms from this chapter, and check your answers in the glossary:
inter-VLAN routing, SVI, FIB, adjacency table, packet rewrite

Use Command Reference to Check Your Memory

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

To test your memory of the inter-VLAN routing and CEF configuration and verification commands, use a piece of paper to cover the right side of Tables 11-3 and 11-4, respectively. Read the description on the left side, and then see how much of the command you

can remember. Remember that the CCNP exam focuses on practical or hands-on skills that are used by a networking professional.

Table 11-3 *Inter-VLAN Routing Configuration Commands*

Task	Command Syntax
Put a port into Layer 2 mode.	Switch(config-if)# switchport
Put a port into Layer 3 mode.	Switch(config-if)# no switchport
Define an SVI.	Switch(config)# interface vlan <i>vlan-id</i>

Table 11-4 *Multilayer Switching Verification Commands*

Task	Command Syntax
Show a Layer 2 port status.	Switch# show interface <i>type member/module/number</i> switchport
Show a Layer 3 port status.	Switch# show interface <i>type member/module/number</i>
Show an SVI status.	Switch# show interface vlan <i>vlan-id</i>
View the FIB contents.	Switch# show ip cef
View FIB information for an interface.	Switch# show ip cef [<i>type member/module/number</i> vlan <i>vlan-id</i>] [detail]
View FIB information for an IP prefix.	Switch# show ip cef [<i>prefix-ip prefix-mask</i>] [longer-prefixes] [detail]
View FIB adjacency information.	Switch# show adjacency [<i>type member/module/number</i> vlan <i>vlan-id</i>] [summary detail]
View counters for packets not switched by CEF.	Switch# show cef not-cef-switched