Ethernet LAN switching

This chapter covers

- The definition of a LAN
- The contents of the Ethernet header and trailer
- How switches learn the MAC addresses of devices in the network
- How switches forward frames to the appropriate destination
- How network hosts use ARP to learn the MAC address of other hosts
- The ping utility

In this chapter, we will cover Ethernet LAN switching, which is the process switches use to forward frames to their proper destinations within a LAN. A frame is a Layer 2 PDU, including the Layer 2 header, trailer, and payload; we covered PDUs in chapter 4. When a network host sends a frame out of its port, it is the switch's role to make sure the frame reaches its proper destination.

This chapter covers material from domain 1.0 of the CCNA exam topics: Network Fundamentals. Specifically, we will cover the following topics:

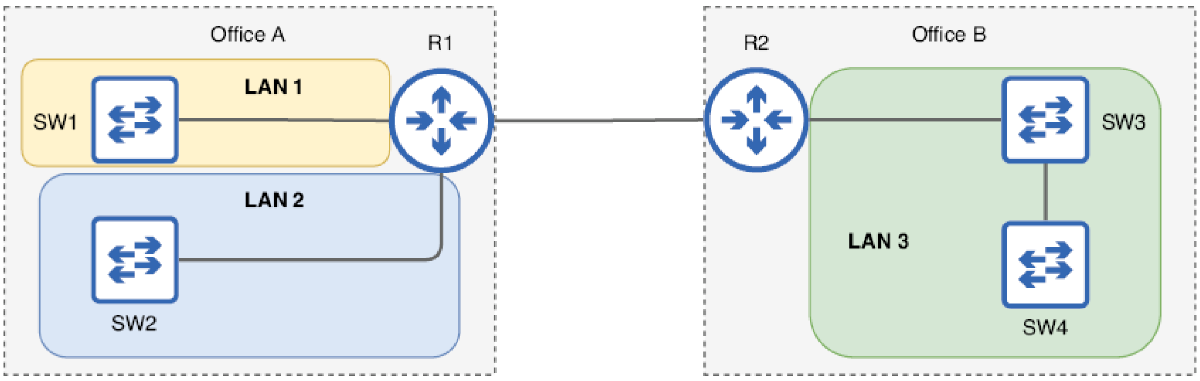
- 1.13 Describe switching concepts
  - 1.13a MAC learning and aging
  - 1.13b Frame switching
  - 1.13c Frame flooding
  - 1.13d MAC address table

It is often said that switches are Layer 2 devices or that they operate at Layer 2. The reason for this is that switches use information in the Layer 2 header (the Ethernet header) to make forwarding decisions. This is in contrast to routers, which use information in the Layer 3 header (the IP header) to make forwarding decisions. We will cover how routers forward network traffic between LANs in part 2 of this book, but for now, we will focus on how switches forward traffic within a LAN.

6.1 Local area networks

1.16 Frame flooding

Figure 6.1 Two offices with two switches in each office. In Office A each switch is a separate LAN because the switches are connected via a router. In Office B both switches are in the same LAN because they are directly connected to each other.



It is often said that switches are *layer 2 devices* or that they *operate at layer 2*. The reason for this is that switches use information in the layer 2 header (the Ethernet header) to make forwarding decisions. This is in contrast to routers, which use information in the layer 3 header (the IP header) to make forwarding decisions. We will cover how routers forward network traffic between LANs in part 2 of this book, but for now we will focus on how switches forward traffic within a LAN.

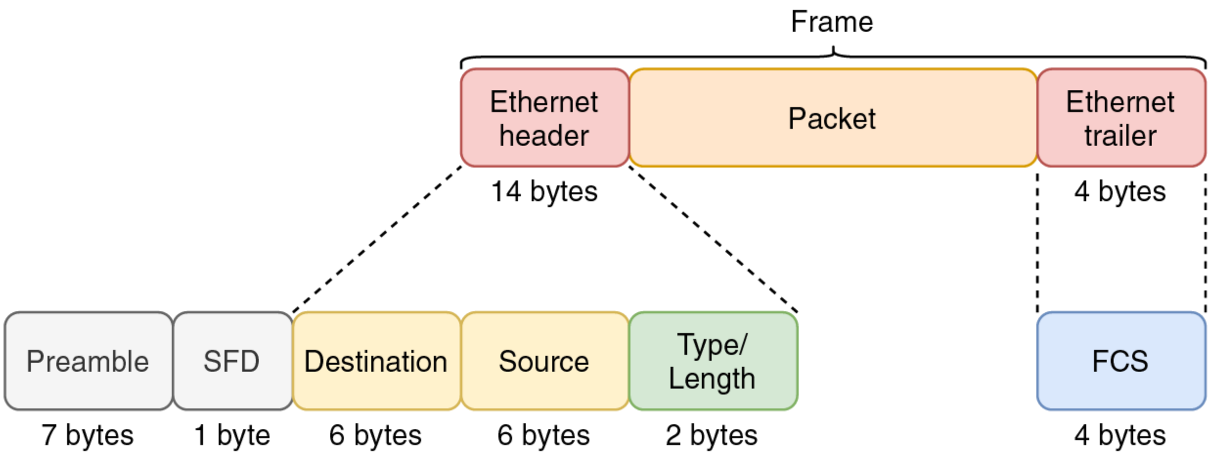
6.1 Local Area Networks

In chapter 2 I defined a local area network (LAN) as a group of interconnected devices in a limited area such as an office, and stated that the role of a switch is to connect devices within a LAN. The precise definition of a LAN can vary depending on the context, but for the purpose of this lesson, how the devices are connected is more significant than the actual physical distance between them. Figure 6.1 demonstrates this concept.

6.2 The Ethernet header and trailer

There are two offices in the diagram, so you might say there are two LANs, and that would not be incorrect if defining a LAN only by physical location. However, in Office A the two switches are not directly connected to each other; each is connected to a different port on R1, and the purpose of a router is to provide connectivity *between* LANs. So, each switch in Office A is its own LAN. For end hosts connected to SW1 to communicate with end hosts connected to SW2, their messages must pass through R1, because it separates the two LANs.

Figure 6.2 The contents of the Ethernet header and trailer. They are split into multiple fields, each serving a different purpose. The fields of the header are the Destination, Source, and Type/Length. The trailer consists of a single field: the Frame Check Sequence (FCS). Although not considered part of the Ethernet frame, the Preamble and SFD are sent with each frame.



6.2.1 Preamble and SFD

6.2 The Ethernet header and trailer

Switches make forwarding decisions using information in the Ethernet header, so to understand switching it's important to understand the contents of that header (and trailer). Figure 6.2 shows the structure of an Ethernet frame. Note that the *Preamble* and *Start Frame Delimiter* (SFD) are included in the diagram but not considered part of an Ethernet frame. We will examine why shortly.

6.2.2 Destination and source

The Preamble and *Start-Frame Delimiter* (SFD) are sent before each Ethernet frame to allow the receiving device to synchronize its *receiver clock* and prepare to receive the incoming frame. This *clock* has nothing to do with the date and time, but rather with how the receiving device interprets the incoming electrical signals – the receiving device needs to determine the precise length of one bit.

Definition

The reason the Preamble and SFD are not considered part of the Ethernet frame, although they are sent with each frame, is that they are purely a function of layer 1, the Physical layer. They do not contain information that influences what the receiving device decides to do with the frame. As mentioned in previous chapters, Ethernet includes specifications at both layer 1 and layer 2, but the layer 1 aspects of Ethernet are not considered part of a frame, which is a layer 2 concept.

6.2.2 Destination & Source

These two fields are perhaps the most significant of the Ethernet header and trailer; the *Destination field* is the destination *MAC address* of the frame, and the *Source field* is the source MAC address of the frame.

The hexadecimal number system

A *Media Access Control address* (MAC address) is a type of address used by layer 2 protocols such as Ethernet and Wi-Fi. MAC addresses are 6 bytes in length and are typically written as a series of 12 hexadecimal characters. They are assigned by the device's manufacturer and should be globally unique.

Layer 2 provides hop-to-hop delivery of messages, and MAC addresses enable that; at layer 3, the message is addressed to the IP address of the final destination host, but at layer 2 the message is addressed to the MAC address of the next hop. Within a LAN, it is a switch's job to look at the destination MAC address of the frame and forward it to the appropriate destination. The source MAC address field is also important because it helps the switch learn which port each host is connected to (more about that in a few pages).

Table 6.1 Decimal numbers and their hexadecimal equivalents (view table figure)

Dec.	Hex.		Dec.	Hex.		Dec.	Hex.		Dec.	Hex.
0	0		8	8		16	10		24	18
1	1		9	9		17	11		25	19
2	2		10	A		18	12		26	1A
3	3		11	B		19	13		27	1B
4	4		12	C		20	14		28	1C
5	5		13	D		21	15		29	1D
6	6		14	E		22	16		30	1E
7	7		15	F		23	17		31	1F

Note

The number system we typically use in our daily lives is the *decimal* number system, which uses 10 digits to represent all values: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Hexadecimal is a number system that uses 16 digits; it uses the same 10 digits used in the decimal system, and borrows 6 letters from the alphabet: A, B, C, D, E, and F.

Because more digits are available, hexadecimal can express large values in fewer characters. In decimal, to express the value after 9 we have to add another character – it becomes 10, a 1 and a 0. Hexadecimal can express that same value in a single character: A. The efficiency of hexadecimal over decimal becomes more significant as the values become greater. Table 6.1 lists some decimal numbers and their equivalent hexadecimal numbers.

Characteristics of MAC addresses

Note

You can use a prefix to indicate whether a number is a decimal or hexadecimal number: *0d* for decimal and *0x* for hexadecimal. This can be useful in networking because we use multiple systems: binary, decimal, and hexadecimal. *10* in decimal is ten, but *10* in hexadecimal is equal to 16 in decimal. To clearly differentiate between the two, you can write *0d10* or *0x10*.

In part 6 of this book (IPv6) we will practice converting between decimal, hexadecimal, and binary. For now that is not necessary; it is enough to understand that MAC addresses are typically written in hexadecimal.

Characteristics of MAC addresses

We have already covered two characteristics of MAC address: they are 6 bytes in length and are usually written in hexadecimal. By writing them in hexadecimal we can express the address in fewer characters; MAC addresses are written as 12 hexadecimal characters, rather than 48 bits (ones and zeros). The details regarding how those 12 characters are notated can vary. Below is a single MAC address written using three different notational conventions. Of course, because this is a CCNA book I will follow Cisco's convention for writing MAC addresses, but it's worth knowing that they can be written in other ways. For comparison, I have also included the address written in binary – I'm sure you'll agree that the hexadecimal representations are easier to read.

0cf5.a452.b101 (used by Cisco IOS)

0c:f5:a4:52:b1:01 (used by macOS)

000011001111010110100100010100101011000100000001 (binary).

Unlike IP addresses (which we'll cover in chapter 7), MAC addresses are not assigned by the network admin or engineer configuring the device. Instead, each port of a network device has a MAC address that is assigned to it by the manufacturer. For this reason, another name for a MAC address is a *burned-in address* (BIA): it is *burned into* the physical port. A MAC address is globally unique – it should not be shared by a port on any other device in the world.

Note

It is possible to override a manufacturer-assigned MAC address with manual configuration, but it is extremely rare to do so.

To ensure that MAC addresses remain globally unique, the first half of each MAC address (the first 3 bytes) is an *organizationally unique identifier* (OUI) assigned to the manufacturer by the IEEE. Then, the manufacturer is free to use the second half to assign unique MAC addresses to each device they manufacture. For example, the MAC addresses of the first three ports of the Cisco switch in my home network are:

Let's summarize MAC addresses before moving on:

- 0cf5.a452.b102
- 0cf5.a452.b103
- *0cf5.a4* is Cisco's OUI (actually Cisco has many OUIs), and the second half is a unique identifier for each port on the switch. As you probably noticed, those three MAC addresses are quite similar – only the final digit is different. That's because MAC addresses on the same device are typically assigned sequentially.
- Let's summarize MAC addresses before moving on:

- MAC addresses are 6-byte (48-bit) addresses assigned to ports by the device's manufacturer. Another name for a MAC address is *burned-in address* (BIA).

The first 3 bytes are an organizationally unique identifier (OUI), assigned to the manufacturer by the IEEE.

Note

MAC addresses are written as 12 hexadecimal characters.

6.2.3 Type/Length

The Type/Length field is a 2-byte field that can be used either to indicate the type of the encapsulated packet (for example, an IP version 4 packet or an IP version 6 packet), or to indicate the length of the encapsulated packet (in bytes). There are historical reasons why this field can be used for two purposes, but both uses are now officially part of the Ethernet standard. These days, in almost all cases this field is used to indicate the type of the encapsulated packet; instead of the length being indicated by this field, the end of the frame is indicated by a special signal after the frame.

Note

The original IEEE 802.3 standard used this field exclusively to indicate the length of the encapsulated packet, and an additional header was used to indicate the type of encapsulated protocol: the Logical Link Control (LLC) header, sometimes with an additional Subnetwork Access Protocol (SNAP) extension to that header. However, this is beyond the scope of the CCNA exam.

A value of 1536 or greater in this field indicates the *type* of the encapsulated packet, which is usually IP version 4 (IPv4) or IP version 6 (IPv6). When used to indicate the type of the encapsulated packet, this field is called the *EtherType* field. For reference, here are the values in this field for IPv4 and IPv6, both of which are significant topics on the CCNA exam (usually hexadecimal notation is used; I'm including the decimal numbers for comparison):

6.2.4 Frame Check Sequence

FCS is the name of the field, but the name for this kind of checksum is *cyclic redundancy check* (CRC). The term *cyclic* refers to the kind of algorithm used to calculate the checksum. *Redundancy* means that the field is *redundant* – it expands the size of the message but doesn't add any additional information. *Check* is self-explanatory – it is used to check if the frame traveled from source to destination without the data being corrupted.

IPv6: 0x86DD (0d34525)

Note

Values between 1500 and 1536 should not be used in this field.

6.3 Frame switching

The *Frame Check Sequence* (FCS) is the only field of the Ethernet trailer. It is 4 bytes in length and is used to detect corrupted data in the frame. Before a device sends a frame, it uses an algorithm to calculate a *checksum*, a small block of data that is appended to the end of the frame as the FCS field.

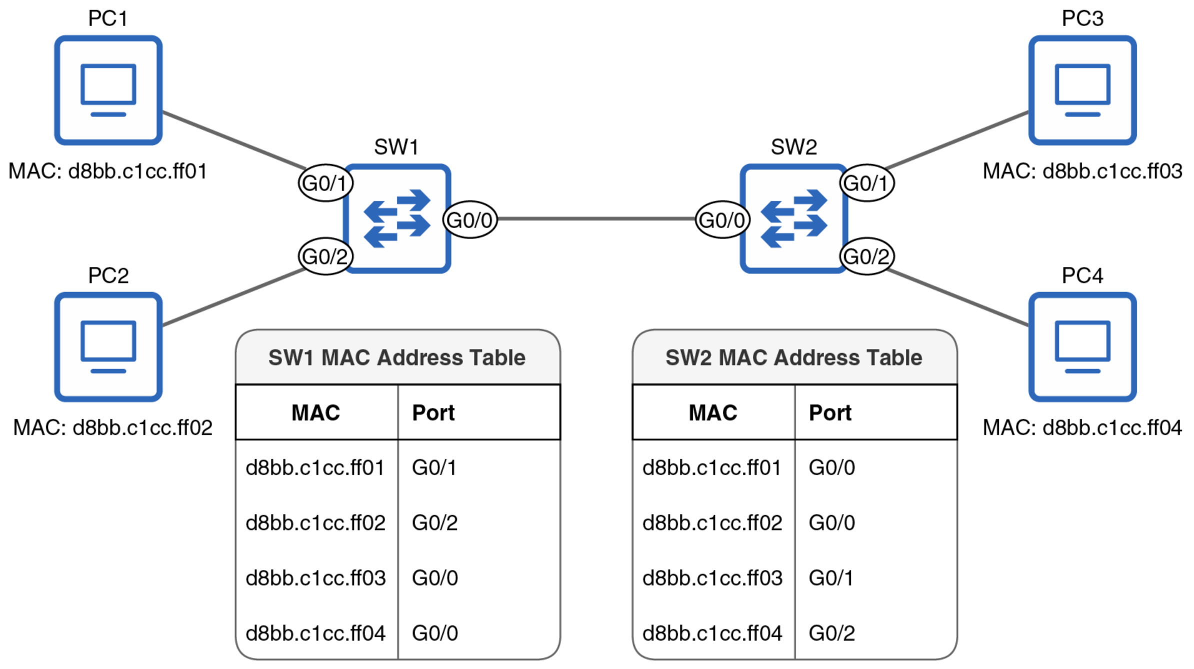
FCS is the name of the field, but the name for this kind of checksum is *cyclic redundancy check* (CRC). The term *cyclic* refers to the kind of algorithm used to calculate the checksum. *Redundancy* means that the field is *redundant* – it expands the size of the message but doesn't add any additional information. *Check* is self-explanatory – it is used to check if the frame traveled from source to destination without the data being corrupted.

Note

Now that we have looked at the information in the Ethernet header and trailer, let's see how switches use the Source and Destination fields to build a *MAC address table* and forward frames to the appropriate destination(s) within a LAN.

6.3.1 MAC address learning

Figure 6.3 A network with two switches, each with two PCs connected. SW1 and SW2 have learned the MAC address of each PC by examining the Source field of frames received on their ports as the PCs communicate with each other. SW1 knows it can reach PC1 via its G0/1 port, PC2 via its G0/2 port, and PC3/PC4 via its G0/0 port. SW2 knows it can reach PC1/PC2 via its G0/0 port, PC3 via its G0/1 port, and PC4 via its G0/2 port.



SW1 MAC Address Table

MAC	Port
d8bb.c1cc.ff01	G0/1
d8bb.c1cc.ff02	G0/2
d8bb.c1cc.ff03	G0/0
d8bb.c1cc.ff04	G0/0

SW2 MAC Address Table

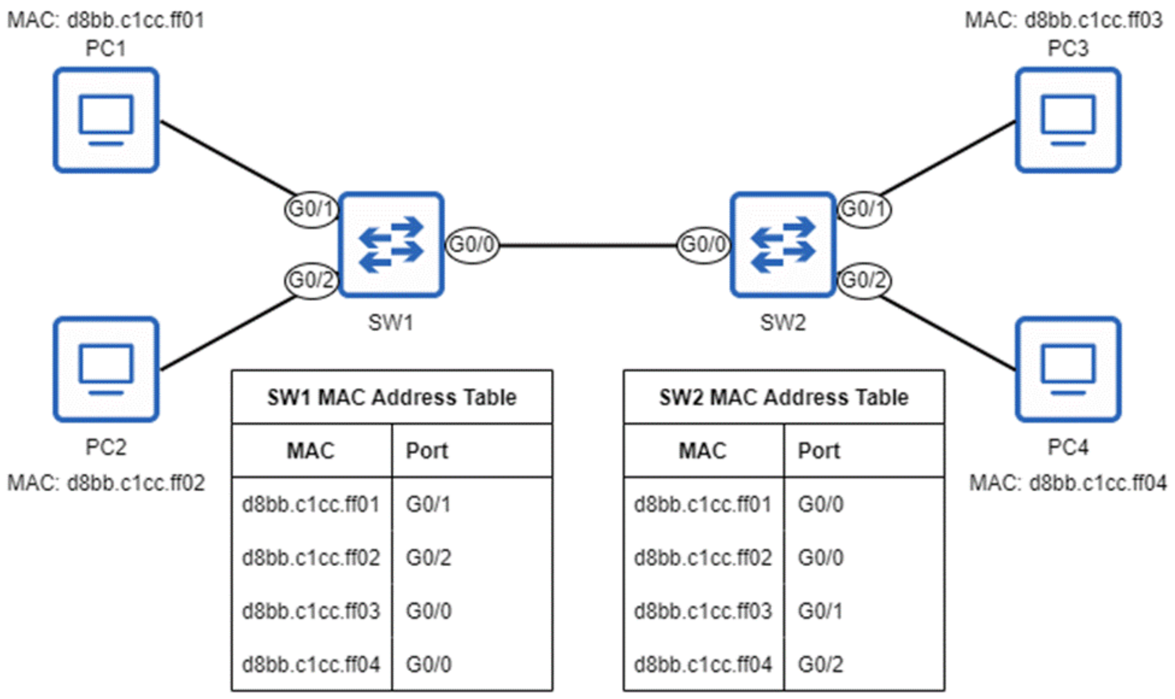
MAC	Port
d8bb.c1cc.ff01	G0/0
d8bb.c1cc.ff02	G0/0
d8bb.c1cc.ff03	G0/1
d8bb.c1cc.ff04	G0/2

Vgieur 6.3 owssh vru attes le brk network arfte uxr switches xzou elnreda qvr MAC addresses vl xrb dseivec jn xur ZXG. Hvoewer, xw skt simsign c wxl eiscep el roq zeluzp, asdh cz wxg switches frdoraw iftarfc borefe hrgv coeq lbuit irhet WYX address table z yzn eqw rqk LRa alern vcga roseht' MAC addresses.

Port names on Cisco devices

MAC addresses learned by a switch in this manner are known as *dynamic* MAC addresses – they are automatically (*dynamically*) learned. This is in contrast to *static* MAC addresses, which are manually (*statically*) configured, although as stated above that is quite rare. A switch will remove a dynamic MAC address from its MAC address table after five minutes of inactivity (if it doesn't receive a frame from that MAC address for five minutes); this is called *MAC aging*.

Figure 6.3 A network with two switches, each with two PCs connected. SW1 and SW2 have learned the MAC address of each PC by examining the Source field of frames received on their ports as the PCs communicate with each other. SW1 knows it can reach PC1 via its G0/1 port, PC2 via its G0/2 port, and PC3/PC4 via its G0/0 port. SW2 knows it can reach PC1/PC2 via its G0/0 port, PC3 via its G0/1 port, and PC4 via its G0/2 port.



SW1 MAC Address Table

MAC	Port
d8bb.c1cc.ff01	G0/1
d8bb.c1cc.ff02	G0/2
d8bb.c1cc.ff03	G0/0
d8bb.c1cc.ff04	G0/0

SW2 MAC Address Table

MAC	Port
d8bb.c1cc.ff01	G0/0
d8bb.c1cc.ff02	G0/0
d8bb.c1cc.ff03	G0/1
d8bb.c1cc.ff04	G0/2

Port Names on Cisco Device

6.3.2 Frame flooding and forwarding

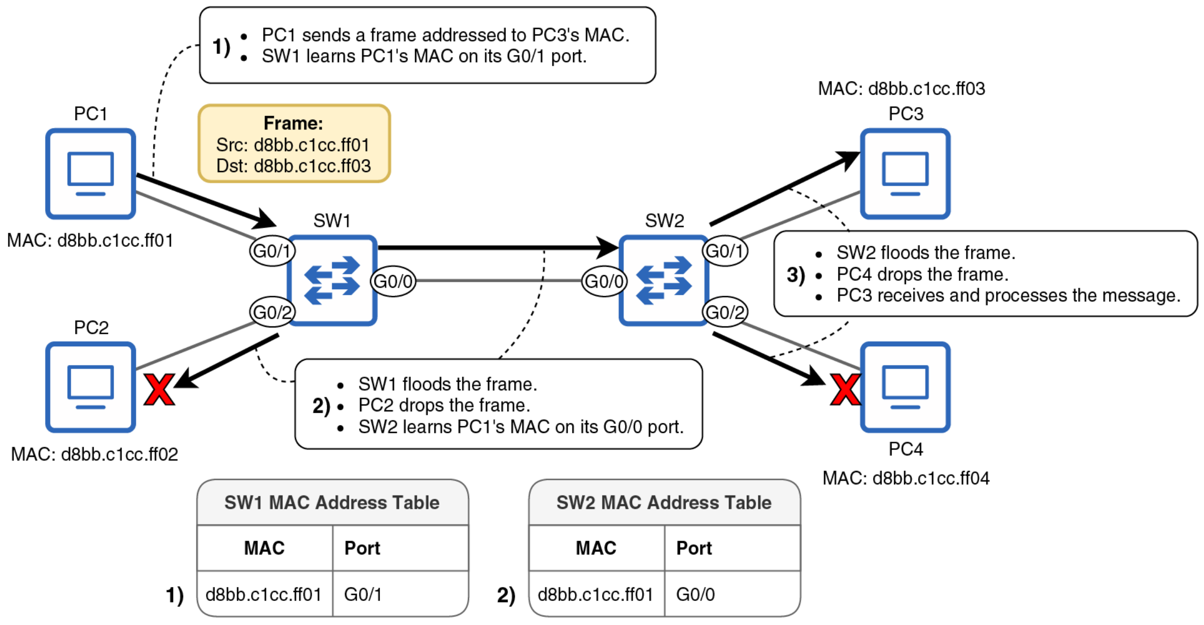
In this book I will use a two number system (*X*/*Y*), in which the first number is the *slot* on the device and the second number is the port number within that slot. A *slot* is a group of ports on a network device. In many cases the ports in a slot are *modular*, meaning you can insert *modules* with different kinds of ports depending on your needs. Additionally, I will shorten the names to use the first letter only: E = Ethernet, F = FastEthernet, G = GigabitEthernet, T = TenGigabitEthernet.

Once the switches have learned the MAC address of each host in the LAN, as in figure 6.3, forwarding traffic is simple – when a switch receives a frame, it looks up the destination MAC address in its MAC address table and forwards the frame out of the appropriate port. For example, if PC1 sends a frame to PC2's MAC address, SW1 will check its MAC address table and see that it should forward the frame out of its G0/2 port. This frame from PC1 is a *known unicast* frame.

Definition

A frame addressed to a single destination host is called a *unicast* frame. If the switch already has an entry for the frame's destination MAC address in its MAC address table, it is called a *known unicast* frame.

Figure 6.4 PC1 sends a unicast frame to PC3, but neither SW1 nor SW2 have an entry for the destination MAC address in their MAC address table. (1) PC1 sends the frame, and SW1 learns PC1's MAC address. (2) SW1 floods the frame. SW2 learns PC1's MAC address. PC2 drops the frame. (3) SW2 floods the frame. PC4 drops the frame. PC3 receives and processes it.


1) • PC1 sends a frame addressed to PC3's MAC.
• SW1 learns PC1's MAC on its G0/1 port.
PC1
MAC: d8bb.c1cc.ff01
Frame:
Src: d8bb.c1cc.ff01
Dst: d8bb.c1cc.ff03
SW1
G0/1
G0/0
G0/2
PC2
MAC: d8bb.c1cc.ff02
2) • SW1 floods the frame.
• PC2 drops the frame.
• SW2 learns PC1's MAC on its G0/0 port.
SW2
G0/0
G0/1
G0/2
MAC: d8bb.c1cc.ff03
PC3
3) • SW2 floods the frame.
• PC4 drops the frame.
• PC3 receives and processes the message.
PC4
MAC: d8bb.c1cc.ff04

SW1 MAC Address Table

	MAC	Port
1)	d8bb.c1cc.ff01	G0/1

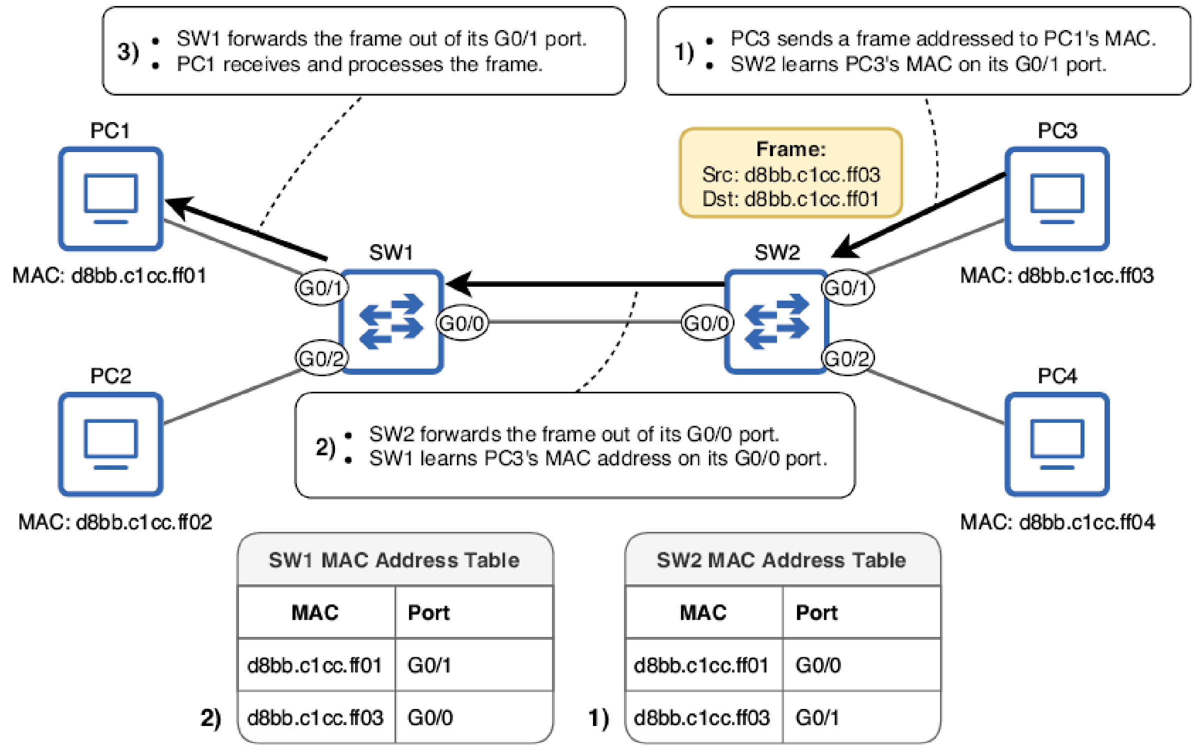
SW2 MAC Address Table

	MAC	Port
2)	d8bb.c1cc.ff01	G0/0

PC1 sends a unicast frame addressed to PC3's MAC address. SW1 uses the Source of the frame to learn PC1's MAC address, and then it *floods* the frame – it sends the frame out of every port except the one it was received on (G0/1). SW1 doesn't have an entry for the PC3's MAC address in its MAC address table, so by flooding the frame it hopes the frame will be able to reach PC3, and then it will later be able to learn PC3's MAC address when PC3 sends a reply.

Definition

Figure 6.5 PC3 replies to PC1's message. (1) PC3 sends the frame, and SW2 learns PC3's MAC address. (2) SW2 forwards the frame out of its G0/0 port, and SW1 learns PC3's MAC address. (3) SW1 forwards the frame out of its G0/1 port, and PC1 receives and processes it.


3) • SW1 forwards the frame out of its G0/1 port.
• PC1 receives and processes the frame.
1) • PC3 sends a frame addressed to PC1's MAC.
• SW2 learns PC3's MAC on its G0/1 port.
PC1
Frame:
Src: d8bb.c1cc.ff03
Dst: d8bb.c1cc.ff01
PC3
MAC: d8bb.c1cc.ff01
SW1
SW2
MAC: d8bb.c1cc.ff03
G0/1
G0/0
G0/0
G0/1
G0/2
G0/2
PC2
PC4
2) • SW2 forwards the frame out of its G0/0 port.
• SW1 learns PC3's MAC address on its G0/0 port.
MAC: d8bb.c1cc.ff02
MAC: d8bb.c1cc.ff04

SW1 MAC Address Table

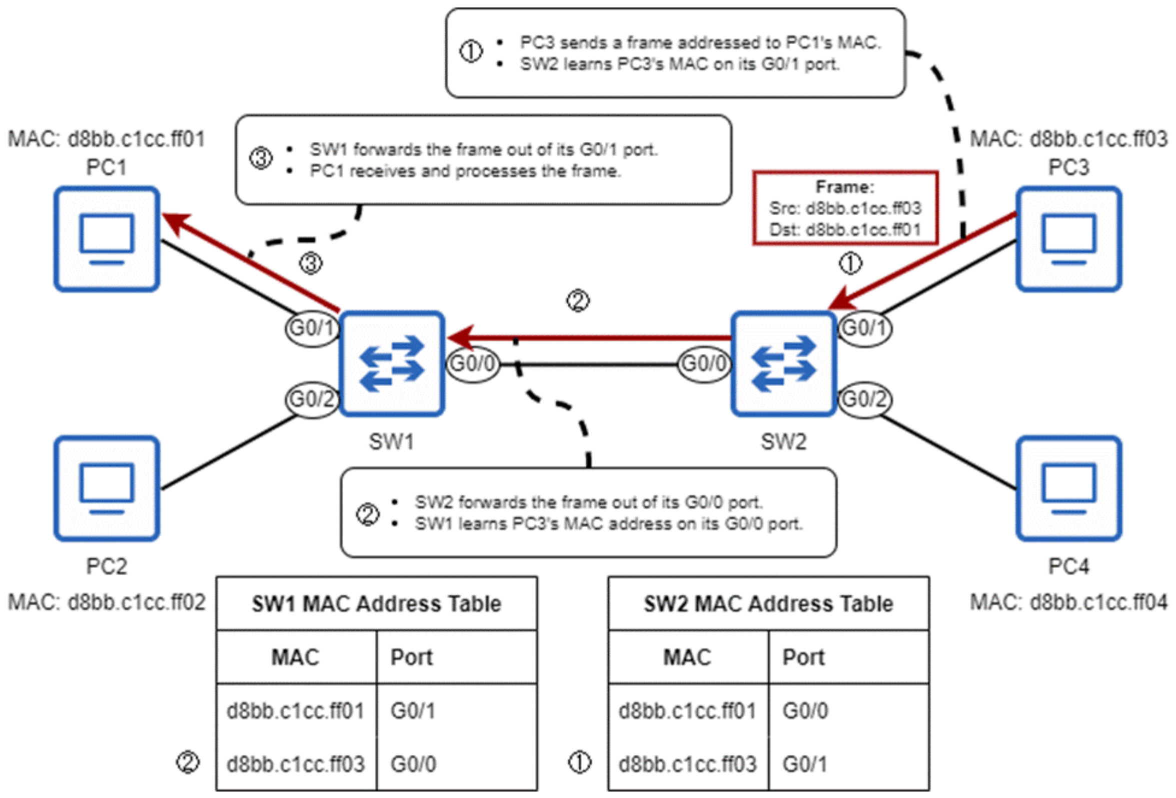
	MAC	Port
	d8bb.c1cc.ff01	G0/1
2)	d8bb.c1cc.ff03	G0/0

SW2 MAC Address Table

	MAC	Port
	d8bb.c1cc.ff01	G0/0
1)	d8bb.c1cc.ff03	G0/1

When SW1 floods the frame, both PC2 and SW2 receive it. Because the destination MAC address of the frame is not PC2's, it drops the frame. SW2, on the other hand, will treat the frame just like SW1 did; it will learn PC1's MAC address and then flood the frame out of its G0/1 and G0/2 ports.

- When SW2 floods the frame, both PC3 and PC4 receive it. Like PC2, PC4 will drop the frame because the destination MAC address is not its own. However, PC3 sees that the frame is destined for its own MAC address, so PC3 will receive and process the message. Figure 6.5 shows what then happens when PC3 sends a reply back to PC1.
- **Figure 6.5 PC3 replies to PC1's message. 1) PC3 sends the frame, and SW2 learns PC3's MAC address. 2) SW2 forwards the frame out of its G0/0 port, and SW1 learns PC3's MAC address. 3) SW1 forwards the frame out of its G0/1 port, and PC1 receives and processes it.**



① • PC3 sends a frame addressed to PC1's MAC.
• SW2 learns PC3's MAC on its G0/1 port.

② • SW2 forwards the frame out of its G0/0 port.
• SW1 learns PC3's MAC address on its G0/0 port.

③ • SW1 forwards the frame out of its G0/1 port.
• PC1 receives and processes the frame.

SW1 MAC Address Table

	MAC	Port
	d8bb.c1cc.ff01	G0/1
②	d8bb.c1cc.ff03	G0/0

SW2 MAC Address Table

	MAC	Port
	d8bb.c1cc.ff01	G0/0
①	d8bb.c1cc.ff03	G0/1

Note

Known unicast frame: *forward*

Unknown unicast frame: *flood*

6.3.3 The MAC address table in Cisco IOS

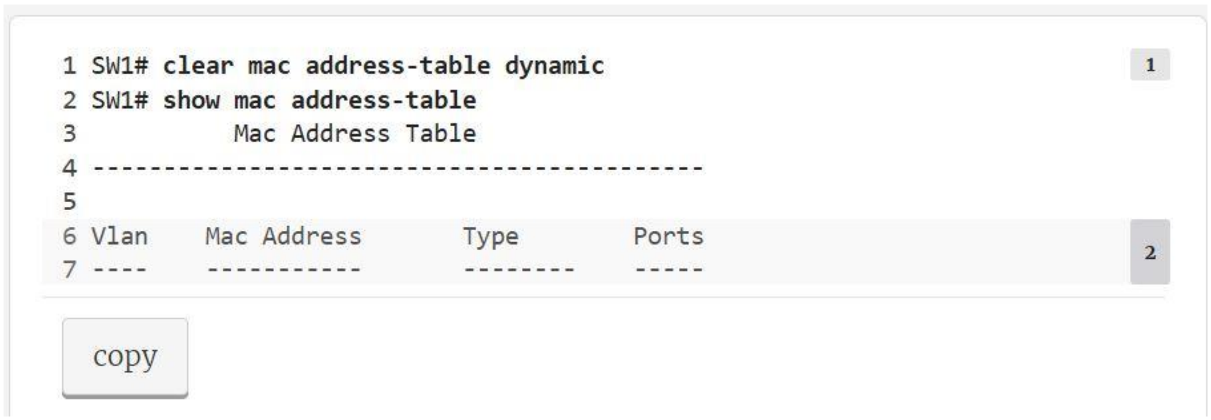
```
SW1# show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    5254.0017.7cd2    DYNAMIC     Gi0/0
   1    d8bb.c1cc.ff01    DYNAMIC     Gi0/1
   1    d8bb.c1cc.ff02    DYNAMIC     Gi0/2
   1    d8bb.c1cc.ff03    DYNAMIC     Gi0/0
   1    d8bb.c1cc.ff04    DYNAMIC     Gi0/0
```

Note

Cisco abbreviates GigabitEthernet ports as “GiX/X,” not “GX/X.”

6.3.3 The MAC address table in Cisco IOS

The command to view a Cisco switch's MAC address table is show mac address-table (in user EXEC or privileged EXEC mode). As the following example shows, there are a few more columns than just the MAC address and port. The Type column indicates whether the MAC address was dynamically learned (DYNAMIC) or statically configured (STATIC). The Vlan column indicates which *virtual LAN* (VLAN) each MAC address was learned in. We will cover VLANs in chapter 12. For now just note that all of the MAC addresses are in VLAN 1 by default.



```
SW1# clear mac address-table dynamic
SW1# show mac address-table
          Mac Address Table
-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
```

Note

Although you can usually leave a switch to learn MAC addresses itself and clear them as needed (after five minutes of inactivity), you can manually clear dynamic MAC addresses from a switch's MAC address table with the clear mac address-table dynamic command. The example below shows this; I clear SW1's MAC address table and then view it again, but it is empty.

6.4 Address Resolution Protocol

You can also specify a specific address to remove from the MAC address table, or tell the switch to remove all MAC addresses learned on a specific port. To clear a specific dynamic MAC address from the table you can use the clear mac address-table dynamic address mac-address command. To clear all dynamic MAC addresses learned on a specific interface, use the clear mac address-table dynamic interface interface-name command. However, as stated previously you usually will not have to manually interfere with a switch's MAC address learning and aging processes. Note that this command uses the term *interface* instead of *port*. As you will see when we cover more configurations, this is true of most commands within Cisco IOS.

6.4 Address Resolution Protocol

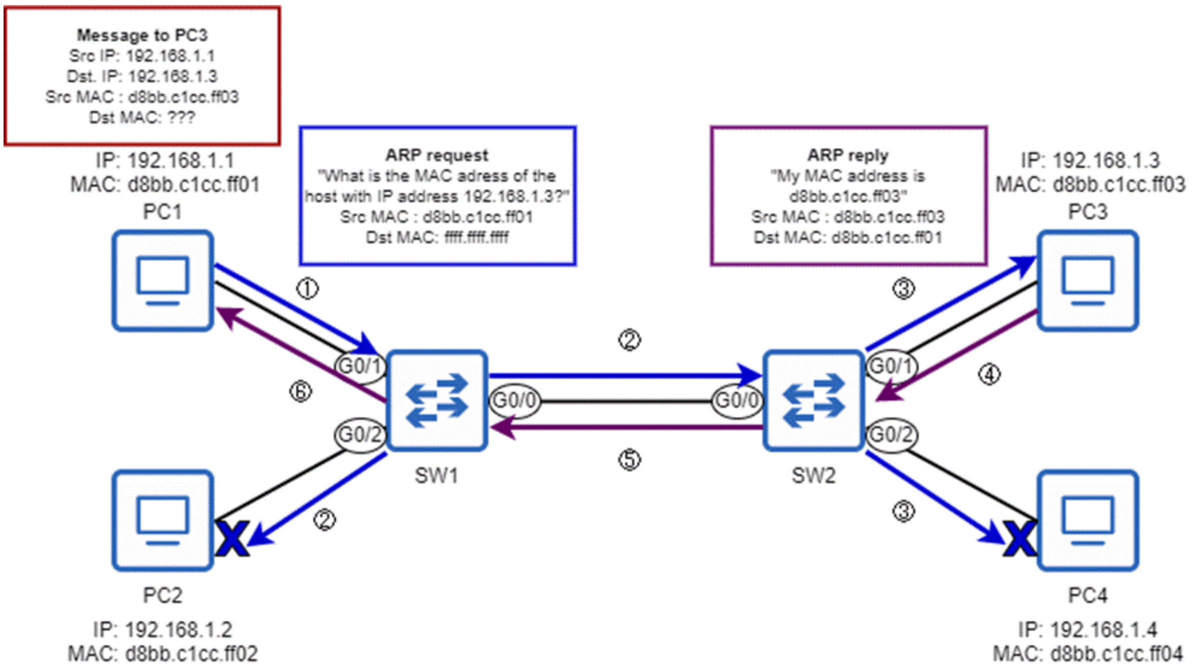
Now we have looked at how switches forward frames and learn the MAC addresses of devices in their LAN. Next we will fill in another piece of the puzzle – how the PCs know each other's MAC addresses. To do so, they use *Address Resolution Protocol* (ARP).

Figure 6.6 An ARP request and reply exchange between PC1 and PC3. PC1 wants to send a message to PC3 but does not know PC3's MAC address, so PC1 uses ARP to learn PC3's MAC address.

Here's the process shown in figure 6.6:

The IP addresses of PC1, PC2, PC3, and PC4 are shown in figure 6.6, but it's not necessary to understand the structure of IP addresses yet. We will cover IP addresses in the next chapter.

Figure 6.6 An ARP request and reply exchange between PC1 and PC3. PC1 wants to send a message to PC3, but does not know PC3's MAC address. 1) PC1 sends an ARP request message, addressed to the broadcast MAC (ffff.ffff.ffff). 2) SW1 broadcasts the frame. PC2 sees the ARP request is not for its own IP address, so it drops the message. 3) SW2 floods the message. PC4 sees the ARP request is not for its own IP address, so it drops the message. PC3 sees the ARP request is for its own IP address, so 4) it sends an ARP reply to PC1. 5) SW2 forwards the ARP reply out of its G0/0 port. 6) SW1 forwards the ARP reply out of its G0/1 port. PC1 now knows PC3's MAC address, so it will be able to send the original message to PC3.



Note

The group of devices which will receive a broadcast message sent by one of the group's members are said to be in the same *broadcast domain*. A broadcast domain can be thought of as equivalent to a LAN or layer 2 domain. Devices connected to the same switch (or different switches, but the switches are connected, as in figure 6.6) are in the same broadcast domain.

After the ARP exchange is complete, PC1 knows PC3's MAC address; it will store PC3's MAC address in its *ARP table*, which is a list of IP addresses and their associated MAC addresses. Now PC1 will be able to send its message in a frame addressed to PC3's MAC address. It is also worth mentioning that, upon receiving PC1's ARP request message, PC3 also stores PC1's MAC address in its own ARP table.

Unicast messages can be thought of as *one-to-one* and broadcast as *one-to-all*. Additionally, there is another type of message called *multicast*, which is *one-to-multiple* (but not necessarily all). We will touch on multicast messages in later chapters.

We have covered how switches learn MAC addresses, how they flood and forward frames, and how hosts learn the MAC address of another host in the LAN by sending an ARP request to that host's IP address, but there is still one more part to the puzzle. How does a host know the IP address of the host it wants to send a message to? The answer is "it depends". We will cover some possibilities in this book, for example *Domain Name System* (DNS), which is used to convert hostnames (ie. manning.com) into IP addresses. Another option is that the user of the device could manually specify the IP address to send a message to, such as when using *ping* to test connectivity.

6.5 Ping

Note

To send a ping message to another host on the network, the command is ping ip-address (this is true for Cisco IOS, Windows, Linux, macOS, etc.). The IP address of the destination host is specified directly in the command, so that's how the source host knows the IP address of the destination host.

Ping is a component of the *Internet Control Message Protocol* (ICMP), which plays a supporting role to the *Internet Protocol* (IP). In your networking career (or career in nearly any other area of IT), you'll certainly use ping very frequently as a simple way to test if two hosts can reach each other over the network; it is a very common diagnostic and troubleshooting tool. Like ARP, ping consists of two messages: an *ICMP echo request* and *ICMP echo reply*. However, unlike ARP, both of the messages used by ping are unicast.

```
PC1# ping 10.0.0.2

Type escape sequence to abort.

Sending 5, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:

 !!!!!

Success rate is 100 percent (5/5), round-trip min/avg/max = 3/3/4 ms
```

6.6 Summary

- A *local area network* (LAN) can be defined as a portion of a network where hosts can communicate with each other without the use of a router. This is also called a *layer 2 domain*.
- The Ethernet header has three fields: Destination, Source, and Type/Length. The Ethernet Trailer has one field: the *Frame Check Sequence* (FCS).
- Although they are not considered part of the Ethernet frame, the Preamble and *Start Frame Delimiter* (SFD) are sent with each frame.
- The Preamble is a 7-byte series of alternating binary 1s and 0s. The SFD is a single byte in length and uses the bit pattern 10101011 to indicate the end of the Preamble and the beginning of the frame.

- The Destination and Source fields are each 6 bytes in length and contain the MAC address of the frame's sender (Source) and its intended receiver (Destination). MAC addresses are assigned by the manufacturer and should be globally unique.
- MAC addresses are usually written in hexadecimal, a number system that uses 16 characters: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. When written in hexadecimal, MAC addresses are 12 characters in length.
- The first half of a MAC address is the *organizationally unique identifier* (OUI), which is assigned to the device's manufacturer by the IEEE. The second half of the MAC address can be used freely by the manufacturer to assign a unique MAC address to each port of the devices it manufactures.
- The Type/Length field either indicates the *type* of the encapsulated packet (i.e., IPv4 or IPv6) or the *length* of the encapsulated packet in bytes. If the value is 1500 or less, it indicates the length of the packet. If the value is 1536 or greater, it indicates the type (in this case, the name is *EtherType*).
- The FCS field uses a *cyclic redundancy check* (CRC) to allow the receiving host to check for errors in the frame that might have occurred in transit.
- A switch makes decisions about how to forward frames by looking up each frame's destination MAC address in the switch's MAC address table.
- A switch builds its MAC address table by looking at the source MAC address field of frames it receives and creating an entry in the MAC address table, associating the MAC address with the port the frame was received on. This is called *MAC address learning*.
- MAC addresses learned like this are called *dynamic MAC addresses*.
- If a switch doesn't receive a frame from a dynamic MAC address for 5 minutes, it will remove the entry from the MAC address table. This is called *MAC address aging*.
- A frame addressed to a single host is called a *unicast* frame. If the switch already has an entry for the frame's destination MAC in its MAC address table, it is a *known unicast* frame, and the switch will forward it out of the appropriate port. If the switch does not have an entry for the frame's destination MAC in its MAC address table, it is an *unknown unicast* frame, and the switch will *flood* the frame, sending it out of all ports (except the one it was received on).
- The **show mac address-table** command allows you to view the MAC address table of a Cisco switch. Dynamic MAC addresses can be cleared with **clear mac address-table dynamic**, **clear mac address-table dynamic address** *mac-address*, or **clear mac address-table dynamic interface** *interface-name*.
- Address Resolution Protocol (ARP) allows a host to learn the MAC address of another host in the network. It uses two messages: an *ARP request* and an *ARP reply*.
- The ARP request is sent to the broadcast MAC address (ffff.ffff.ffff), so it is flooded by switches. The ARP reply is a unicast message.

- A *broadcast domain* is a group of devices that receive broadcast messages from each other. Devices connected to the same switch (or different switches, but the switches are connected) are in the same broadcast domain.
- The ARP table is used to store IP address-to-MAC address mappings, so an ARP request doesn't have to be sent before every single packet.
- Ping is a utility that tests connectivity between two network hosts. It is a component of the Internet Control Message Protocol (ICMP) and is a common diagnostic and troubleshooting tool. To send a ping, use the **ping** *ip-address* command.
- Ping uses two messages: an *ICMP echo request* and an *ICMP echo reply*. Both are unicast messages.