CCNA™
CISCO® CERTIFIED NETWORK ASSOCIATE

3

IP Addressing

CERTIFICATION OBJECTIVES

The Transmission Control Protocol/Internet Protocol (TCP/IP) is a standard that includes many protocols. It defines how machines on an internetwork can communicate with each other. It was initially funded by and developed for DARPA (Defense Advanced Research Protects Agency), which is a conglomeration of U.S. military and government organizations. Developed initially for the government, it was later made available to the public, mainly seen on Unix systems. First specified in RFC 791, it has become the defacto standard for networking protocols. The Internet uses TCP/IP to carry data between networks, and most corporations today use TCP/IP for their networks. This chapter will provide an overview of TCP/IP, including some of its more important protocols, as well as IP addressing.

exam Watch

It is VERY important that you understand ALL aspects of IP addressing. Therefore, spend a lot of time on this chapter. If you don't understand IP addressing when taking the exam, you will have a difficult time in passing it. This Exam Watch goes for all three exams: INTRO, ICND, and CCNA.

CERTIFICATION OBJECTIVE 3.01

TCP/IP Protocol Stack

To help articulate how data is moved between devices running TCP/IP, a model was developed that resembles the OSI Reference Model discussed in Chapter 2. Table 3-1 compares the two models. The following sections will cover the layers of the TCP/IP Protocol stack.

Application Layer

One main difference between the OSI Reference Model and TCP/IP's model is that TCP/IP lumps together the application, presentation and session layers into one layer, called the application layer. Here are some common TCP/IP applications Cisco devices support: DNS, HTTP, SNMP, telnet, and TFTP.

TABLE 3-1 Comparison of the OSI Reference Model and the TCP/IP Protocol Stack

Layer	OSI Reference Model	TCP/IP Protocol Stack
Layer 7	Application	
Layer 6	Presentation	
Layer 5	Session	Application
Layer 4	Transport	Transport
Layer 3	Network	Internet
Layer 2	Data Link	Data Link
Layer 1	Physical	Physical

Transport Layer

The TCP/IP transport layer is responsible for providing a logical connection between two devices and can provide these two functions:

- Flow control (through the use of windowing or acknowledgements)
- Reliable connections (through the use of sequence numbers and acknowledgements)

exam Watch

TCP/IP's transport layer can provide for flow control and reliable connections.

The transport layer packages application layer data into *segments* to send to a destination device. The remote destination is responsible for taking the data from these segments and forwarding it to the correct application. TCP/IP has two transport layer protocols: Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). These protocols are discussed in the following sections.

TCP

TCP's main responsibility is to provide a reliable connection-oriented logical service between two devices. It can also use windowing to implement flow control so that a source device doesn't overwhelm a destination with too many segments.

exam Watch

Here are some examples of applications (and their ports) that use TCP: HTTP (80), FTP (21), SMTP (25), and telnet (23).

TCP Segment TCP transmits information between devices in a data unit called a segment. Table 3-2 shows the components of a segment.

TABLE 3-2 TCP Segment Components

TCP Field Name	Length (in bits)	Definition
Source Port	16	Identifies which application is sending information
Destination Port	16	Identifies which application is to receive the information
Sequence Number	32	Maintains reliability and sequencing
Acknowledgement Number	32	Used to acknowledge received information
Header Length	4	Number of 32-bit words that comprise the header
Reserved Field	6	Currently not used (set to all zeroes)
Code Bits	6	Defines control functions, like synchronization
Window Size	16	Indicates the number of segments allowed to be sent before waiting for an acknowledgment from the destination
Checksum	16	CRC of the header and encapsulated application data
Urgent Field	16	Points to the any urgent data in the segment
Options	0-32	
Data		Application data (not part of the TCP header)

The segment is composed of a header, followed by the application data. Without any options, the TCP header is 20-bytes in length.

TCP's Multiplexing Function TCP, and UDP, provide a multiplexing function for a device: This allows multiple applications to simultaneously send and receive data. With these protocols, port numbers are used to differentiate the connections. Port numbers are broken into two basic categories: well-known port numbers (sometimes called reserved port numbers) and source connection port numbers. Each application is assigned a well-known port number that is typically between 1 and 1,023. Any time you want to make a connection to a remote application, your application program will use the appropriate well-known port number.

exam Watch

Be familiar with the TCP field names, especially the fact that a TCP segment contains a sequence and acknowledgment number as well as a window size.

As you saw in Table 3-2, however, there happens to be two port numbers in the segment: source and destination. When you initiate a connection to a remote

application, your operating system will pick a currently unused port number greater than 1,023 and assign this number as the source port number. Based on the application that you are running, the application will fill in the destination port number with the well-known port number of the application. When the destination receives this traffic, it looks at the destination port number and knows which application this traffic should be directed to. This is also true for returning traffic from the destination. This process was discussed in Chapter 2.

Port numbers are assigned by the Internet Assigned Numbers Authority (IANA). When a vendor develops a new commercial application and wants a reserved (well-known) port number, he applies for one to this organization. Here are some common TCP applications with their assigned port numbers: FTP (20 and 21), HTTP (80), SMTP (25), and telnet (23).

TCP's Reliability TCP provides a reliable connection between devices by using sequence numbers and acknowledgements. Every TCP segment sent has a sequence number in it. This not only helps the destination reorder any incoming frames that arrived out of order, but it also provides a method of verifying if all sent segments were received. The destination responds to the source with an acknowledgment indicating receipt of the sent segments.

Before TCP can provide a reliable connection, it has to go through a synchronization phase, called a *three-way handshake*. Here are the steps that occur during this setup process:

1. The source sends a synchronization frame with the SYN bit marked in the Code field. This segment contains an initial sequence number. This is referred to as a SYN segment.
2. Upon receipt of the SYN segment, the destination responds back with its own segment, with its own initial sequence number and the appropriate value in the acknowledgement field indicating the receipt of the source's original SYN segment. This notifies the source that the original SYN segment was received. This is referred to as a SYN/ACK segment.
3. Upon receipt of the SYN/ACK segment, the source will acknowledge receipt of this segment by responding back to the destination with an ACK segment, which has the acknowledgment field set to an appropriate value based on the destination's sequence number.

Here is a simple example of this three-way handshake:

1. Source sends a SYN: sequence number = 1

2. Destination responds with a SYN/ACK: sequence number = 10, acknowledgement = 2
3. Source responds with an ACK segment: sequence number = 2, acknowledgement = 11

exam Watch

TCP uses a three-way handshake to set up a reliable connection: SYN, SYN/ACK, and ACK.

In this example, the destination's acknowledgment (step 2) is one greater than the source's sequence number, indicating to the source that the next segment expected is 2. In the third step, the source sends the second segment, and, within the same segment in the Acknowledgement field, indicates the receipt of the destination's segment with an acknowledgment of 11--one greater than the sequence number in the destination's SYN/ACK segment. This process was described in Chapter 2.

Windowing TCP allows the regulation of the flow of segments, ensuring that one device doesn't flood another device with too many segments. TCP uses a sliding windowing mechanism to assist with flow control. For example, if you have a window size of 1, a device can send only one segment, and then must wait for a corresponding acknowledgement before sending the next segment. If the window size is 20, a device can send 20 segments and then has to wait for an acknowledgment before sending 20 additional segments.

The larger the window size is for a connection, the less acknowledgments that are sent, thus making the connection more efficient. Too small a window size can affect throughput, since a device has to send a small number of segments, wait for an acknowledgment, send another bunch of small segments, and wait again. The trick is to figure out an optimal window size: one that allows for the best efficiency based on the current conditions in the network and on the two devices.

exam Watch

TCP employs a positive acknowledgement with retransmission (PAR) mechanism to recover from lost segments. The same segment will be repeatedly resent, with a delay between each segment, until an acknowledgement is received from the destination. The acknowledgement contains the sequence number of the segment received and verifies receipt of all sent prior segments. This eliminates the need for multiple acknowledgements and resending acknowledgements.

A nice feature of this process is that the window size can be dynamically changed through the lifetime of the connection. This is important because many more connections may come into a device with varying bandwidth needs. Therefore, as a device becomes saturated with segments from many connections, it can, assuming that these connections are using TCP, lower the window size to slow the flow of segments coming into it. TCP windowing is covered in RFC 793 and 813.

UDP

Where TCP provides a reliable connection, UDP provides an unreliable connection. UDP doesn't go through a 3-way handshake to set up a connection--it just begins sending its information. Likewise, UDP doesn't check to see if sent segments were received by a destination; in other words, it doesn't have an acknowledgment process. Typically, if an acknowledgment process is necessary, the transport layer (UDP) won't provide it; instead, the application itself, at the application layer, will provide this verification.

exam Watch

UDP is more efficient that TCP because it has less overhead. Here are some examples of UDP applications, along with their assigned port numbers: DNS queries (53), RIP (520), SNMP (161), and TFTP (69).

Given these deficiencies, UDP does have an advantage over TCP: it has less overhead. For example, if you only need to send one segment, and receive one segment back, and that's the end of the transmission, it makes no sense to go through a 3-way handshake to first establish a connection and then send and receive the two segments: this is not very efficient. DNS queries are a good example where the use of UDP makes sense. Of course, if you are sending a large amount of data to a destination, and need to verify that it was received, then TCP would be a better transport mechanism.

Table 3-3 contains the components of a UDP segment. Examining this table, you can notice a lot of differences between a UDP and TCP segment. First, since UDP

TABLE 3-3 UDP Segment Components

UDP Field Name	Length (in bits)	Definition
Source Port	16	Identifies the sending application
Destination Port	16	Identifies the receiving application
Length	16	Denotes the size of the UDP segment
Checksum	16	Provides a CRC on the complete UDP segment
Data		Application data (not part of the UDP header)

is connectionless, there is no need for sequence and acknowledgment numbers. And second, since there is no flow control, there is no need for a window size field. As you can see, UDP is a lot simpler, and more efficient, than TCP. Any control functions that need to be implemented for the connection are not done at the transport layer--instead, these are handled at the application layer.

Internet Layer

Layer-3 of the TCP/IP protocol stack is called the *Internet* layer. The corresponding layer in the OSI Reference Model is the network layer. The *Internet Protocol* (IP) is just one of the protocols that reside at this layer. It is very common in the industry to hear people refer to TCP/IP as just "IP"; however, this is a misnomer, since IP is just one of many protocols within TCP/IP. Other IP protocols include ARP, RARP, ICMP, OSPF, and others. The next few sections explain the components of an IP packet and some of the protocols that function at the Internet layer.

exam Watch

IP provides a connectionless, unreliable connection to other devices. If reliability and flow control are required, TCP (transport layer) can provide this.

IP Datagram

Where the transport layer uses segments to transfer information between machines, the Internet layer uses datagrams. Datagram is just another word for *packet*. Table 3-4 shows the components of the IP datagram. Without any options, the IP header is 20 bytes in length.

exam Watch

IP is uses a TTL field to limit the number of hops a packet can travel. Here are some common protocols and their protocol numbers: ICMP (1), IGRP (9), IPv6 (41), TCP (6), and UDP (17).

The main function of the IP datagram is to carry protocol information for either Internet layer protocols or encapsulated transport layer protocols. To designate what protocol the IP datagram is carrying in the data field, the IP datagram carries the protocol's number in the Protocol field of the datagram.

ICMP

The *Internet Control Message Protocol* (ICMP) is used to send error and control information between TCP/IP devices. ICMP, defined in RFC 792, includes many different messages that devices can generate or respond to. Here is a list of these messages: Address Reply, Address Request, Destination Unreachable, Echo, Echo Reply, Information Reply, Information Request, Parameter Problem, Redirect, Subnet Mask Request, Time Exceeded, Timestamp, and Timestamp Reply.

TABLE 3-4 IP Datagram Components

IP Field Name	Length (in bits)	Definition
Version	4	IP version number, like IPv4
Header Length	4	Length of the IP header in 32-bit word values
Priority and TOS (Type of Service)	8	Defines how the IP network should treat the datagram
Total Length	16	Length of the IP datagram, including the header and encapsulated data
Identification	16	
Flags	3	Is set if the datagram is a fragment; also used for other purposes
Fragment Offset	13	Defines information about the datagram if it is a fragment
TTL (Time-To-Live)	8	Sets the number of allowed layer-3 hops the datagram is allowed to traverse
Protocol	8	Identifies the protocol (like TCP, UDP, ICMP, OSPF, etcetera) that was used to encapsulate payload information
Header Checksum	16	Checksum on just the IP header fields
Source IP Address	32	IP address of the source device
Destination IP address	32	IP address of the destination device
Options	0-32	
Data		Protocol information (like an encapsulated UDP segment or ICMP information)

exam Watch

Two common applications that use ICMP are ping and traceroute (trace). Ping uses an ICMP echo message to test connectivity to a remote device.

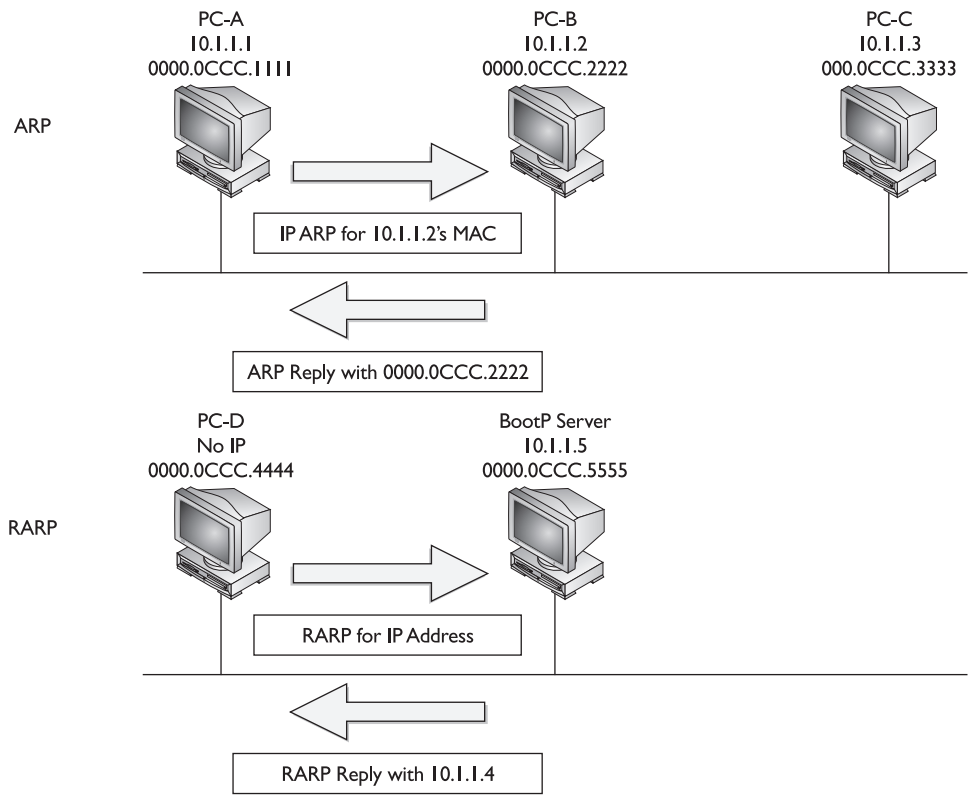
One of the most common implementations using ICMP is ping. Ping uses a few ICMP messages, including echo, echo request, and destination unreachable. Ping is used to test whether or not a destination is available. A source generates an ICMP echo packet. If the destination is available, it will respond back with an echo reply. If it isn't available, a router will respond back with a destination unreachable message. Trace is an application that will list the IP addresses of the routers along the way to the destination, displaying the path the packet took to reach the destination.

ARP and RARP

The *Address Resolution Protocol* (ARP) is an Internet layer protocol that helps TCP/IP devices find other devices in the same broadcast domain. ARP uses a local broadcast to discover neighboring devices. Basically, ARP resolves an IP address of a destination to the MAC address of the destination on the same data link layer medium. Remember that for two devices to talk to each other in Ethernet, the data link layer uses MAC addresses to differentiate the machines on the segment. And that when devices talk to each other at the data link layer, they need to know the destination's MAC address.

The top part of Figure 3-1 shows an example of the use of ARP. In this example PC-A wants to send information directly to PC-B. PC-A knows PC-B's IP address, however, it doesn't know PC-B's Ethernet MAC address. To resolve the IP to MAC address, PC-A generates an IP ARP. In the ARP datagram, the source IP address is 10.1.1.1 and the destination is 255.255.255.255—every device on the segment. PC-A includes PC-B's IP address in the data field of the ARP datagram. This is encapsulated into an Ethernet

FIGURE 3-1 ARP and RARP Examples

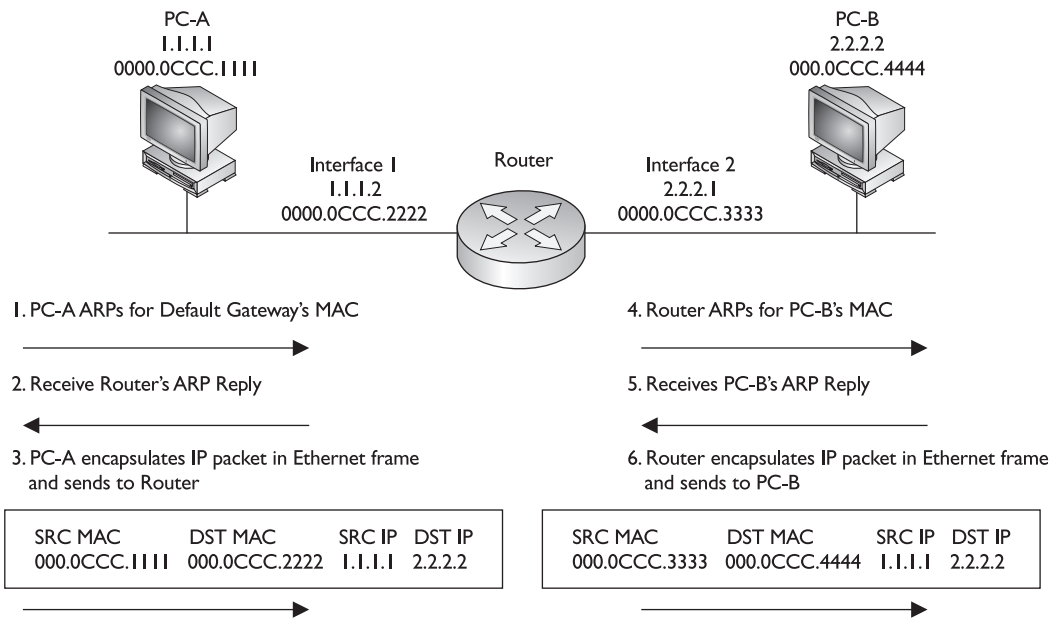


frame, with a source MAC address of 0000.0CCC.1111 and a destination MAC address of FFFF.FFFF.FFFF and is then placed on the wire. Both PC-B and PC-C see this frame. Both devices notice the data link layer broadcast address and assume that this frame is for them, so they pass it up to the Internet layer. Again, there is a broadcast address in the destination IP address field, so both devices examine the data payload. PC-B notices that this is an ARP and that this is its IP address, and therefore responds directly back to PC-A with PC-B's MAC address. PC-C, however, sees that this is not an ARP for its MAC address and ignores the datagram.

Figure 3-2 shows a more detailed example of the use of ARP. In this example, PC-A wants to connect to PC-B using IP. The source address is 1.1.1.1 (PC-A) and the destination is PC-B (2.2.2.2). Since the two devices are on different networks, a router is used to communicate between the networks. Therefore, if PC-A wants to send something to PC-B, it has to be via the intermediate router. This communication does not occur at the network layer using IP; however, it occurs at the data link layer. I'll assume that Ethernet is being used in this example.

The first thing that PC-A will do is to determine if the destination is local to this subnet or on another subnet (I'll discuss this process when I cover IP addressing and subnetting later in this chapter). In this example, it's a remote location, so PC-A will ARP for the default gateway's MAC address--note that one thing you must configure

FIGURE 3-2 ARP Example with a Router



on PC-A, besides it's own IP address and subnet mask, is the default gateway address. This is shown in step 1 of Figure 3-2. In step 2, the router responds back with the MAC address of the interface connected to PC-A. In step 3, PC-A takes the IP packet with the source and destination IP addresses (the source is 1.1.1.1 and the destination is 2.2.2.2) and encapsulates this in an Ethernet frame, with the source MAC address of PC-A and the destination MAC address of the router.

When the router receives the Ethernet frame, it compares the frame to its own MAC address, which it matches. The router strips off the Ethernet frame and makes a routing decision based on the destination address of 2.2.2.2. In this case, the network is directly connected to the router's second interface, which also happens to be Ethernet. In step 4, the router ARPs for the MAC address of 2.2.2.2 (PC-B) and receives the response in step 5. The router then encapsulates the IP packet in an Ethernet frame in step 6, placing its second interface's MAC address, which is sourcing the frame, in the source MAC address field and PC-B's MAC address in the destination field. When PC-B receives this, it knows the frame is for itself (matching destination MAC address) and that PC-A originated the IP packet that's encapsulated).

Note that in this example, the IP packet was not altered by the router, but two Ethernet frames are used to get the IP packet to the destination. Also, each device will keep the MAC addresses in an ARP table, so the next time PC-A needs to send something to PC-B, the devices will not have to ARP each other again.

exam Watch

Be familiar with what device talks to what at both layer-2 and layer-3. With a router between the source and destination, the source, at layer-2, uses its own MAC address as the source but the default gateway MAC address as the destination. Note that the IP addresses used at layer-3 are not changed by the router.

RARP is sort of the reverse of an ARP. In an ARP, the device knows the layer-3 address, but not the data-link layer address. With a RARP, the device doesn't have an IP address and wants to acquire one. The only address that this device has is a MAC address. Common protocols that use RARP are BOOTP and the Dynamic Host Configuration Protocol (DHCP).

The bottom part of Figure 3-1 shows a RARP example. In this example, PC-D doesn't have an IP address and wants to acquire one. It generates a data-link layer broadcast (FFFF.FFFF.FFFF) with an encapsulated RARP request. This examples assumes that the RARP is associated with BOOTP. If there is a BOOTP server on

exam Watch

DHCP allows devices to dynamically acquire their addressing information. This information can include a client IP address and subnet mask, a default gateway, DNS, TFTP, and WINS server addresses, a domain name, and the length of the lease of the client address.

the segment, and if it has an IP address for this machine, it will respond back. In this example, the BOOTP server, 10.1.1.15, has an address (10.1.1.4) and assigns this to PC-D, sending this address as a response to PC-D.

CERTIFICATION OBJECTIVE 3.02

IP Addressing Introduction

Probably one of the most confusing aspects of the TCP/IP protocol stack is the addresses used at the Internet layer, referred to as IP addresses. The remainder of this chapter will focus on IP addressing, its components, and how to plan for addressing. Please note that there are two different versions of TCP/IP: IPv4 and IPv6. Only IPv4 is covered in this book.

IPv4 addresses are 32 bits in length. However, to make the addresses readable, they are broken into four bytes (called octets), with a period (decimal) between each byte. So that the address is understandable to the human eye, the four sets of binary numbers are then converted to decimal. Let's look at a simple example: 11111111111111111111111111111111, which is 32 1's. This is broken up into four octets, like this: 11111111.11111111.11111111.11111111. Then each of these octets are converted into decimal, resulting in 255.255.255.255. The format of this address is commonly called *dotted decimal*.

Bit Values

Before you can begin to understand the conversion process, you need to understand binary mathematics. Computers and networking devices process everything in binary. In a byte (octet), there are eight bits. Each bit, when enabled, represents a specific decimal value. Table 3-5 shows the conversion of a specific bit position when it is enabled. In this table, the bit positions are labeled from left-to-right, where the left-most bit is the most

TABLE 3-5 Binary to decimal conversion for byte values.

Bit Position	8	7	6	5	4	3	2	1
Decimal Value	128	64	32	16	8	4	2	1

significant and the right-most bit is the least-significant. A bit can contain one of two values: 0 or 1. If it is enabled (set to 1), then that equates to a particular decimal value, shown in the second row of Table 3-5. If it is disabled (set to 0), then this equates to a decimal value of zero. Higher-order bits are the ones with a higher-numbered bit position (like 8) while lower-order bits are the ones with a lower-numbered bit position (like 1). To convert the binary byte value to a decimal value, you look at all the bits that are turned on and add up the equivalent decimal values.

exam Watch

Remember how to convert a binary 8-bit value to a decimal number and vice versa.

For example, assume that you had a byte with a value of 11000001. Bits 8, 7, and 1 are on, so add up the associated decimal values to get the corresponding decimal equivalent of the byte value: 128 + 64 + 1 = 193. If you had a byte value of 00110011, the decimal value would be: 32 + 16 + 2 + 1 = 51. If all the bit positions where set to 0, then the decimal value would be 0. If all the bit positions were set to 1, the equivalent decimal value would be: 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255. Given this, a byte value can range from 0 to 255.

Hexadecimal Conversion

Even though IP addressing deals with octal, decimal, and binary notations, you might be required to perform decimal to hexadecimal conversion and vice versa. Therefore, since part of this chapter deals with numeric conversions, I'll briefly cover the process of performing decimal/hexadecimal conversion.

First, as you already know, binary has two possible values in a bit position and octal has 8 bit positions, allowing you to represent numbers from 0-255 in a byte (8 bits). And in decimal, you have values that range from 0-9 (10 values). Hexadecimal has a range of 16 values: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. As an example, a decimal 10 is equivalent to A in hexadecimal. A decimal 17 is equivalent to 10 in hexadecimal. When dealing with hexadecimal, a hex digit is represented in four bits. Table 3-6 lists a handy conversion chart.

exam Watch

You should be familiar with converting binary to both decimal and hexadecimal, as well as hexadecimal to decimal or vice versa.

TABLE 3-6

Binary to Decimal to Hexadecimal Conversion for Bit Values

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

For example, if you had an 8-bit value of 10000001, break this up into two 4-bit values, since a hexadecimal value is represented in 4 bits: 1000 and 00001. In hexadecimal, this value would be 8 and 0, or 80. If you had an 8-bit value of 11011001, this would be D9 in hexadecimal.

Classes of Addresses

Recall from Chapter 2 that logical, or layer-3, addresses, have two components: a network and host number. The network number uniquely identifies a segment in the network and a host number uniquely identifies a device on a segment. The combination of these two numbers must be unique throughout the entire network. TCP/IP uses the same two components for addressing, but does add a twist by breaking

up network numbers into five classes: Class A, B, C, D, and E. Each of these classes has a predefined network and host boundary:

- With a Class A address, the first byte is a network number (8 bits) and the last 3 bytes are for host numbers (24 bits)
- With a Class B address, the first two bytes are a network number (16 bits) and the last 2 bytes are for host numbers (16 bits)
- With a Class C address, the first three bytes are a network number (24 bits) and the last 1 byte is for host numbers (8 bits)
- Class D addresses are used for multicasting and Class E addresses are reserved

exam Watch

Remember the 5 classes of IP addresses, and the fact that Class A addresses have, by default, 8 network bits, Class B 16 bits and Class C 24 bits.

Distinguishing Between Classes of Addresses

Given the above distinction, it would seem that addressing for IP is easy. However, what distinguishes the different classes of addresses are what the first bit to 5 bits is set to:

- Class A addresses always begin with a "0" in the highest order bit
- Class B addresses always begin with "10" in the highest order bits
- Class C addresses always begin with "110" in the highest order bits
- Class D addresses always begin with "1110" in the highest order bits
- Class E addresses always begin with "11110" in the highest order bits

exam Watch

Remember the binary values that IP addresses begin with and be able to determine, by looking at the first binary byte, whether the address is a Class A, B, C, D, or E address.

When talking about the highest-order bit or bits, this includes *all* 32 bits. Therefore, this would be the very first bit on the *left* of the address (the most significant bit). If the first octet contains 1000001, this represents 129 in decimal, which would be a Class B address.

Network Numbers and Classes of Addresses

Given the above distinctions with the assigned high-order bit values, it is easy to predict, for a given address, what class of network numbers it belongs to:

- Class A addresses range from 1-126: 0 is reserved and represents all IP addresses; 127 is a reserved address and is used for testing, like a loopback on an interface: 00000001-01111111.
- Class B addresses range from 128-191: 10000000-10111111.
- Class C addresses range from 192-223: 11000000-11011111.
- Class D addresses range from 224-239: 11100000-11101111.
- Class E addresses range from 240-254: 255 is a reserved address and is used for broadcasting purposes.

Given the above restrictions with beginning bit values, it is fairly easy to predict what address belongs to what class.

exam Watch

Class A addresses range from 1-126, Class B from 128-192, Class C from 192-223, Class D from 224-239 and Class E from 240-254. 127 is reserved for the loopback interface (internal testing). Also remember the ranges in binary.

When you are dealing with IP addresses, there are always two numbers reserved for a given network number: the first address in the network represents the network's address, and the last address in the network represents the broadcast address for this network, commonly called a *directed broadcast*. When you look at IP itself, there are two IP addresses reserved: 0.0.0.0 (the very first address), which represents all IP addresses, and 255.255.255.255 (the very last address), which is the local broadcast address (all devices should process this datagram).

exam Watch

Remember the list of private networks, which cannot be used in public networks: 10.0.0.0, 172.16.0.0-172.31.0.0, and 192.168.0.0-192.168.255.0.

Within this range of addresses for Class A, B, and C addresses, there are some reserved addresses, commonly called *Private Addresses*. All the other addresses in these classes are called public addresses. Anyone can use private addresses; however, this creates a problem if you want to access the Internet. Remember that each device in the network (in this case, this includes the Internet) must have a unique IP address. If two networks are using the same private addresses, then you would run into reachability issues. In order to access the Internet, your source IP addresses must

have a unique Internet public address. This can be accomplished through address translation. Here is a list of private addresses, which are assigned in RFC 1918:

- Class A: 10.0.0.0-10.255.255.255 (1 Class A network)
- Class B: 172.16.0.0-172.31.255.255 (16 Class B networks)
- Class C: 192.168.0.0-192.168.255.255 (256 Class C networks)

Private and public addresses, as well as address translation, are discussed in Chapter 14.

IP Address Components

As was mentioned earlier, there are two components to addressing: network and host. The host portion is actually broken into three subcomponents: network address, host addresses, and directed broadcast address.

The very first address in a network number is called the network address, or *wire number*. This address is used to uniquely identify one segment from all of the other segments in the network. The last address in the network number is called the directed broadcast address, and is used to represent all hosts on this network segment. A directed broadcast is similar to a local broadcast. The main difference is that routers will not propagate local broadcasts, but can propagate directed broadcasts. Any address between the network address and the directed broadcast address is a host address for the segment. You use these middle addresses to assign to host devices on the segment, like PCs, servers, routers, and switches.

exam Watch

Each network has two reserved addresses: a network number (the first address) and a directed broadcast (the last address). Any addresses between these two values can be assigned to networking devices on the segment.

Network and Directed Broadcast Addresses

When dealing with a network address, all of the host bits in the host portion of the address are set to zeros. If all of the host bits in a network number are set to ones, making it the very last address, then this is the directed broadcast address. Any combination of bit values between these two numbers in the *host* portion of the address is considered a host address.

As example, 192.1.1.0 is a Class C address and is also a network number. If you recall from earlier in this chapter, the Class C addresses range from 192-223 in the

first byte and the network number is three bytes long. Therefore, "192.1.1" is the network number. The last byte is the host address. This byte is 0, which is the very first address in the network. Therefore, the network address is 192.1.1.0. If you would set the last 8 bits to all ones (the host bits), which is equivalent to 255 in decimal, this would be the directed broadcast (192.1.1.255) for the network.

Host Addresses

Any number between the network address and the directed broadcast address is a host address. In the previous example, any number between 0 and 255 is a host address for the network 192.1.1.0: 192.1.1.1-192.1.1.254.

An important item to point out about this process is that for any given network number, you *lose* two addresses. The first address in a network is reserved for the network itself and the last address is reserved for the directed broadcast address. There is a formula that defines the number of available host addresses, assuming that you know the number of bits that are reserved for host numbers: $2^N - 2$. At the beginning of this formula, **2** is raised to the power of **N**, where **N** is the number of host bits. When figuring out a power of a number. You take the number and multiply it by itself based on the power it is being raised to. As an example, 2^4 would be 2 * 2 * 2 * 2 = 16. The "**- 2**" part of the formula represents the loss of the first and last addresses. Table 3-7 represents the powers of 2, up to 32.

So, as an example, a Class C network has a 24-bit network number component and an 8-bit host component. Therefore, for a Class C network, the lowest address in this fourth octet is 0 and the last address in this octet is 255 (all 8 bits are set to 1). All numbers between 1-254, are host addresses for the class C network. Using the addressing formula, you can easily show a Class C network has 254 host addresses:

TABLE 3-7 Powers of 2

$2^1 = 2$	$2^9 = 512$	$2^{17} = 131,072$	$2^{25} = 33,554,432$
$2^2 = 4$	$2^{10} = 1,024$	$2^{18} = 262,144$	$2^{26} = 67,108,864$
$2^3 = 8$	$2^{11} = 2,048$	$2^{19} = 524,288$	$2^{27} = 134,217,728$
$2^4 = 16$	$2^{12} = 4,096$	$2^{20} = 1,048,576$	$2^{28} = 268,435,456$
$2^5 = 32$	$2^{13} = 8,192$	$2^{21} = 2,097,152$	$2^{29} = 536,870,912$
$2^6 = 64$	$2^{14} = 16,384$	$2^{22} = 4,194,304$	$2^{30} = 1,073,741,824$
$2^7 = 128$	$2^{15} = 32,768$	$2^{23} = 8,388,608$	$2^{31} = 2,147,483,648$
$2^8 = 256$	$2^{16} = 65,536$	$2^{24} = 16,777,216$	$2^{32} = 4,294,967,296$

exam Watch

Remember that each network loses two addresses for host assignments. Also remember the $2^N - 2$ formula, which you can use, based on number of host bits, to determine the number of hosts a network will support.

$2^N - 2 = 2^8 - 2 = 256 - 2 = 254$. For a Class B network, the number of host addresses is 65,534: $2^N - 2 = 2^{16} - 2 = 64{,}536 - 2 = 65{,}534$. And for a Class A network, the number of host addresses is 16,777,214: $2^N - 2 = 2^{24} - 2 = 16{,}777{,}216 - 2 = 16{,}777{,}214$.

CERTIFICATION OBJECTIVE 3.03

Subnetting

One of the problems with the original IP addressing scheme was that for Class A and B networks, address efficiency was an issue. In other words, how many hosts can you physically put on a network segment? Even with the advent of VLANs, this number did not increase dramatically. With IP, you can get between 300-500 devices in a single broadcast domain before experiencing broadcast problems. This is 1-2 Class C networks. If you would assign a Class B network for this broadcast domain, you'd be wasting over 65,000 addresses.

To overcome this deficiency issue, subnetting was introduced. Subnetting allows you to take some of the higher-order ***host*** bits in a network number and use them to create more networks. In the process of creating more networks, each of these additional networks has a lesser number of hosts. These smaller networks are commonly called *subnets*. One disadvantage of subnetting is that you are losing more addresses--each of these subnets has a network and host address. However, the advantage of subnetting is that you now can more efficiently use your addressing.

Let's look at an example. A Class C network has 8 host bits, giving you a total of 256 addresses. Of these 256 addresses, you can only use 254 for host devices, like PCs, routers, and servers. Let's assume that you use the highest-order bit to create more networks, leaving 7 bits for host addresses. With this example, you are creating two subnets: $2^1 = 2$. In this formula, the 1 is the number of subnet bits. In each of these subnets you have 126 host addresses: $2^7 - 2 = 126$. Originally, you lost 2 addresses in a Class C network. Now that you have two subnets, you are losing a total of 4 addresses. However, you now have two networks.

For example, you might have two segments in your network with 100 hosts each on them. You could assign a separate Class C network to each of these segments, but this would be a very inefficient use of your addresses. By using subnetting, you can more efficiently use your addresses. In this example, one Class C network, subnetted with one subnet bit, created two subnet bits with 126 host addresses each. In this example, you are wasting a smaller number of addresses.

Subnet Masks

TCP/IP is unique amongst most layer-3 addressing schemes. When dealing with TCP/IP addresses, there are actually three components to the address: A network component, a host component, and a *subnet mask*. The function of the subnet mask is to differentiate between the network address, the host addresses, and the directed broadcast address. Subnetting is defined in RFC 950.

Like an IP address, the subnet mask is 32 bits long. In binary, a 1 in a bit position in the subnet mask represents a network component and a 0 in a bit position represents a host component. One restriction of subnet masks is that all the network bits (1s) must be contiguous and all the host bits (0s) are contiguous. This is true not only in a single octet, but across all the bits in all four octets. A subnet mask of 11110000.00001111.11111111.11111111 (240.31.255.255) would be invalid since all the 1s are not contiguous. A subnet mask of 1111111.11111111.11111111.11111000 (255.255.255.248), however, is valid.

There are actually four methods that you can use to represent a subnet mask. Here is a list with a demonstration using a Class C network:

- Dotted-decimal: 192.168.1.0 255.255.255.0
- Number of networking bits: 192.168.1.0/24
- Hexadecimal: 192.168.1.0 0xFFFFFF00
- Binary: 192.168.1.0 11111111111111111111111100000000

The most common of these formats is the dotted-decimal and number of networking bits. The last two are not commonly used.

exam Watch

Subnet mask values, binary 1s and 0s, must be contiguous in order to be considered as a valid subnet mask. When representing subnet masks, be very familiar with both the dotted decimal and number of networking bits nomenclature.

Subnet Masks Values

Given the fact that subnet mask values must have all 1's contiguous and all 0's contiguous, Table 3-8 shows some valid decimal numbers for subnet masks in an octet.

For a Class A network, the default subnet mask is 255.0.0.0: the first octet is the network number and the last three octets are the host numbers. For a Class B network, the default subnet mask is 255.255.0.0: the first two octets are the network number and the last two octets are the host numbers. For a Class C network, the default subnet mask is 255.255.255.0: the first three octets are the network number and the last octet is the host numbers.

One important item to point out is that the subnet mask, in and of itself, means nothing without the context of the IP address associated with it. For example, most people would assume that when you see a subnet mask of 255.255.255.0, you are dealing with a Class C address. However, remember that you can perform subnetting on any class address. So this mask can also be used for Class A and B addresses. Therefore, the IP address and subnet mask have a symbiotic relationship. The following sections will show you the valid subnet mask values for Class A, B, and C networks.

Subnet Masks for A-Class Networks

Table 3-9 shows valid subnet masks for Class A networks. In this table, the number of networking bits is the total number of bits used in networking, including both the network and subnet bits. This is also true with Tables 3-10 and 3-11.

Subnet Masks for B-Class Networks

Table 3-10 shows valid subnet masks for class B networks.

Subnet Masks for C-Class Networks

Table 3-11 shows valid subnet masks for class C networks.

exam Watch

You will need to be very familiar with subnet masks for a given address and the number of networks that a subnet mask creates and the number of host addresses for each network.

As you can see from Tables 3-9, 3-10, and 3-11, you can't just choose any subnet mask and apply it to any class of addresses: Some masks are valid for some classes, but not valid for others. For instance, 255.255.0.0 is a valid mask for Class A and B networks, but is an *invalid* mask for Class C networks.

TABLE 3-8 Valid Subnet Mask Values

00000000 = 0	11100000 = 224	11111100 = 252
100000000 = 128	11110000 = 240	11111110 = 254
110000000 = 192	11111000 = 248	11111111 = 255

TABLE 3-9 Class A Subnets

Subnet Mask	Networking Bits	Number of Networks	Number of Hosts
255.255.255.252	/30	4,194,304	2
255.255.255.248	/29	2,097,152	6
255.255.255.240	/28	1,048,576	14
255.255.255.224	/27	524,288	30
255.255.255.192	/26	262,144	62
255.255.255.128	/25	131,072	126
255.255.255.0	/24	65,536	254
255.255.254.0	/23	32,768	510
255.255.252.0	/22	16,384	1,022
255.255.248.0	/21	8,192	2,046
255.255.240.0	/20	4,096	4,094
255.255.224.0	/19	2,048	8,190
255.255.192.0	/18	1,024	16,382
255.255.128.0	/17	512	32,766
255.255.0.0	/16	256	65,534
255.254.0.0	/15	128	131,070
255.252.0.0	/14	64	262,142
255.248.0.0	/13	32	524,286
255.240.0.0	/12	16	1,048,574
255.224.0.0	/11	8	2,097,150
255.192.0.0	/10	4	4,194,302
255.128.0.0	/9	2	8,388,606
255.0.0.0	/8	1	16,777,216

TABLE 3-10 Class B Subnets

Subnet Mask	Networking Bits	Number of Networks	Number of Hosts
255.255.255.252	/30	32,768	2
255.255.255.248	/29	8,192	6
255.255.255.240	/28	4,096	14
255.255.255.224	/27	2,048	30
255.255.255.192	/26	1,024	62
255.255.255.128	/25	512	126
255.255.255.0	/24	256	254
255.255.254.0	/23	128	510
255.255.252.0	/22	64	1,022
255.255.248.0	/21	32	2,046
255.255.240.0	/20	16	4,094
255.255.224.0	/19	8	8,190
255.255.192.0	/18	4	16,382
255.255.128.0	/17	2	32,764
255.255.0.0	/16	1	65,534

TABLE 3-11 Class C Subnets

Subnet Mask	Networking Bits	Number of Networks	Number of Hosts
255.255.255.252	/30	64	2
255.255.255.248	/29	32	6
255.255.255.240	/28	16	14
255.255.255.224	/27	8	30
255.255.255.192	/26	4	62
255.255.255.128	/25	2	126
255.255.255.0	/24	1	254

exam Watch

When subnetting, depending on the device, the first and last subnet in a network, referred to as** subnet 0, **might or might not be valid. For the exam, remember this, since the exam doesn't tell you one way or the other. However, when looking for an answer, you'll never see both as a valid answer, either the answer will include the first and last subnet or it won't.

CERTIFICATION OBJECTIVE 3.04

Planning IP Addressing

When it comes to addressing, dealing with protocols like AppleTalk, IPX, and XNS is easy: each has a distinct network and host component. With these protocols, there is no such thing as a subnet mask, which can change the boundary between network and host numbers. When I started out with TCP/IP, one of the most difficult tasks I've faced in my networking career was tackling and understanding how to handle subnetting and IP addressing. To make matters worse, IP addressing has its roots in binary mathematics, since this is how computing devices deal with numbers. And considering that I have a degree in Mathematics, and that I had trouble with IP addressing, imagine how strange IP addressing must be to the layman?

Through my years of experience dealing with TCP/IP and teaching Cisco-related courses, I've developed a six-step approach to help students plan for their IP addressing needs in their networks. Here are the six steps:

1. Figure out network and host requirements
2. Satisfy host and network requirements
3. Figure out the subnet mask
4. Figure out the network addresses
5. Figure out the directed broadcasts for your networks
6. Figure out the host values for your networks

The following sections will cover the six steps in depth.

Step 1: Figure Out Network and Host Requirements

In this step, you need to do two things:

- Determine the number of hosts that do, or will, exist on the largest segment in your network.
- Determine the maximum number of segments that you have in your network--this will tell you how many networks, or subnets, you'll need.

If you already are dealing with an existing network, then you have a lot of analysis ahead of you. You'll need to perform the above two tasks, counting hosts on each segment, and the number of segments that you have. Remember that when you are counting hosts, each device with a connection to the segment needs to be counted—this includes PCs, servers, routers, servers, printers, and other devices. Remember that a segment could be used in a logical sense, like all the ports off of a switch, or a VLAN. Switching is discussed in more depth in Chapter 7 and VLANs in Chapter 8. You might even want to leave some room for growth by taking your final numbers and adding to them.

To assist with the remaining 5 steps, I'll create an imaginary network. This network has 14 segments and the largest segment has 14 devices on it. You've been assigned a single class C network number (192.168.1.0). Now you're ready to proceed to step 2.

Step 2: Satisfy Host and Network Requirements

In the second step, you'll use three formulas:

1. 2^X => number of networks you need (**X** represents subnet bits)
2. $2^Y - 2$ => number of hosts on your largest segment (**Y** represents host bits)
3. **X + Y** <= total number of host bits

In the first step, you need to figure out how many bits you need to steal from the host bits to create your subnets. In the second step, you need to figure out how many host bits you need to accommodate your host requirements. And last, you need to make sure that when you add up the bits that you stole for subnets, and the bits that you need for your hosts, that you didn't exceed the original number of host bits that you started out with, based on the class A, B, or C network.

As an example, if you had a Class C network and were subnetting it and needed 5 bits for subnets and 4 bits for hosts, this would total 9 bits. Unfortunately, Class C networks only have 8 host bits to begin with, so this wouldn't work. In this situation,

exam Watch

Remember that the exam might not allow you to use subnet 0. Therefore, in Step 1 above, you might need to subtract 2 from the total valid of networks in order to come up with a valid value. In this chapter, I'm assuming that subnet 0 is valid in all of the examples.

you would either need a Class B network or 2 Class C networks. As an other example, if you had the same Class C network and were subnetting it, and you needed 3 bits for subnets and 4 bits for host addresses, this would total 7 bits. In this situation, the Class C network as 8 bits, and you only need 7. This gives you some flexibility--you could use the extra bit to either create more subnets, or to have more hosts with your 3 bits of subnets.

Let's go back to our original example of 192.168.1.0, where you need 14 subnets with a maximum of 14 hosts on each:

1. 2^X => 14 subnets; in this example, **X** needs to be 4, which would result in 16 subnets.
2. $2^Y - 2$ => 14 hosts; in this example, **Y** needs to be 4, which would result in 14 hosts.
3. **X + Y** <= 8 (class C network); in this example 4 + 4 is less than or equal to 8.

Let's break this down step-by-step. In the first step, you need to find a power of 2 that will give a number that is either equal to or greater than the number of subnets that you need. In our example, the power of 2 needs to be 4: $2^4 = 16$. This meets our subnet requirements, since we only need 14 subnets (there are only 14 segments).

Next, you need to figure out your hosts bits by using this formula: $2^Y - 2$ => 14 required hosts; where **Y** is the necessary number of host bits. In this example, $2^4 - 2 = 14$, so you need 4 host bits to get your required 14 hosts.

And last, since we are dealing with a class C network, we only have 8 original host bits. We need to make sure that the total of our subnetting and host bits does not exceed this original value. In our case, 4 + 4 = 8, so we're okay. If the number of bits totaled higher than 8, then we would need two Class C networks, or a Class B network. If the number of bits were less than 8, then we could allocate the extra bit or bits to either create more subnets and/or hosts. Remember that if you are ever in this situation where you have extra bits to deal with, then you need to closely examine your network and figure out, based on future growth, whether you should create more subnets, or allow for more hosts on a subnet.

Step 3: Figure Out the Subnet Mask

Now that the hardest part is over, the rest of the four tasks is easy. At this point, you now know the number of subnet bits you need. However, when dealing with networking and subnet masks, a subnet mask's network portion contains both network *and* subnet bits. Here's a reminder of the default number of networking bits for a class address: A is 8, B is 16, and C is 24.

Given this, just add the class address bits to the subnet bits, and this gives you the total number of *networking* bits. In our example, this would be 24 + 4 = 28. To make the remaining three steps easier, I recommend that you convert the number of bits of the subnet mask to a dotted decimal mask. Figures 3-8, 3-9, and 3-10 have the lists of subnet masks if you need help. However, this is not too hard of a process. First, remember that a subnet mask, just like an IP address, is represented in a dotted decimal format, where there are 8 bits in each octet. That means, for a Class C mask, the first 24 bits are set to 1. In other words, the mask at least begins with 255.255.255. Our job is to figure out the mask in the last octet. Remember that the four highest bits are for subnetting, so just add up these decimal values: 128 + 64 + 32 + 16 = 240.

exam Watch

Remember how to convert a binary subnet mask value to a dotted-decimal format, like in the above example.

There is actually a short cut that I always like to use. If you recall from our example, the number of host bits that are used are the four lower-order bits. Add up these values: 1 + 2 + 4 + 8, which equals 15. The largest number represented by a byte is 255. Since we're not using these bits, just subtract this value from 255, which will give us the mask value in this byte 255 - 15 = 240. I find it easier to add up the small values and subtract them from 255 than to add up the larger bit-decimal values. Eventually, you wont' have to do this mathematical trick as you become accustomed to performing IP addressing and dealing with subnetting. Going back to our example, our subnet mask for network 192.168.1.0 is **255.255.255.240**, or 192.168.1.0/28.

Step 4: Figure Out the Network Addresses

In step 4, we need to figure out the networks that we created with our new subnet mask. Since IP addressing is done in binary, network addresses will always increment in a multiple of something. We'll use this to our advantage when figuring out what our network numbers are for our Class C network. Remember that the network number has all of the host bits set to 0s.

Actually, we already know what this multiplier is: we figured this out in the second part of step 2, using the $2^4 - 2 = 14$ formula. The 14 value is the number of valid host values for a subnet; however, this is *not* the total number of addresses for the subnet. The subnet also has a network and broadcast address, which is the reason the formula subtracts 2 since you can't use these addresses for host devices. Therefore, in our example, each network has a total of 16 addresses, and is incremented by 16 from subnet-to-subnet.

There is another method of verifying your multiplying value. In a byte, you can have numbers ranging from 0 - 255, resulting in a total of 256 numbers. For this verification, take the subnet mask decimal value in the interesting octet and subtract it from 256. The interesting octet is the octet that contains the network and host boundary. In our case, this is the fourth octet. Therefore, using this trick, 256 - 240 = 16. When you compare this number to the number in the last paragraph, you can be assured that you have done your math correctly.

Now that you have figured out the multiplier, write the very first network down, and then start adding 16 to the interesting octet. Table 3-12 lists the subnet numbers for 192.168.1.0. In this table, notice something interesting concerning the last subnet: 192.168.1.240. The network number in the last octet matches the interesting octet in our subnet mask (240). This will *always* be true when you perform subnetting.

There is one important item to point out about subnetting. In the original RFC for subnetting, you were not allowed to use the first and last subnet. For instance, in our example, we would not be able to use 192.168.1.0/28 and 192.168.1.240/28. However, today, assuming that your TCP/IP protocol stack supports subnet 0 (this refers to these two subnets--first and last), you can. You need to make sure, though, that each device on the segment that will have one of these subnets supports this function. In today's age, this shouldn't be an issue. However, I typically use this subnet for addressing only networking devices, like management devices, which typically support subnet 0.

TABLE 3-12

Network numbers for 192.168.1.0.

192.168.1.0	192.168.1.64	192.168.1.128	192.168.1.192
192.168.1.16	192.168.1.80	192.168.1.144	192.168.1.208
192.168.1.32	192.168.1.96	192.168.1.160	192.168.1.224
192.168.1.48	192.168.1.112	192.168.1.176	192.168.1.240

Step 5: Figure Out the Directed Broadcast Addresses

After figuring out all of your subnets, you next need to figure out what the directed broadcast address is for each subnet. This is very simple. The directed broadcast of a subnet is *one number less* than the next network number. Also, the broadcast address has all of its hosts bits set to binary 1s. Table 3-13 shows our network numbers and directed broadcast addresses. For the last table entry, the directed broadcast address will be the highest possible value in a byte: 255.

exam Watch

As a shortcut, remember that the directed broadcast address is one number less than the network address of the next network number.

TABLE 3-13 Network and Directed Broadcast Addresses for 192.168.1.0/28

Network Addresses	Mathematics	Directed Broadcast Addresses
192.168.1.0	16 - 1	192.168.1.15
192.168.1.16	32 - 1	192.168.1.31
192.168.1.32	48 - 1	192.168.1.47
192.168.1.48	64 - 1	192.168.1.63
192.168.1.64	80 - 1	192.168.1.79
192.168.1.80	96 - 1	192.168.1.95
192.168.1.96	112 - 1	192.168.1.111
192.168.1.112	128 - 1	192.168.1.127
192.168.1.128	144 - 1	192.168.1.143
192.168.1.144	160 - 1	192.168.1.159
192.168.1.160	176 - 1	192.168.1.175
192.168.1.176	192 - 1	192.168.1.191
192.168.1.192	208 - 1	192.168.1.207
192.168.1.208	224 - 1	192.168.1.223
192.168.1.224	240 - 1	192.168.1.239
192.168.1.240		192.168.1.255

Step 6: Figure Out the Host Addresses

Step 6 is the easiest step. If you recall, any address between the network and directed broadcast address is a host address for a given network. We can then complete the rest of our addressing for 192.168.1.0, as is shown in Table 3-14. If you look at the very first subnet in this table, 192.168.1.0, you'll see that it has a total of 14 host addresses, which matches are formula: $2^{Y} - 2$, $2^{4} - 2 = 14$ hosts.

For the CCNA Exam, you will need to understand how to do IP addressing. Of course, on the job, you can cheat and use an IP subnet calculator. One of my favorites is from a company called *Boson Software*. They offer a free download of their subnet calculator (http://www.boson.com), but it is also included on the CD that comes with this book. Boson's subnet calculator will even do route summarization, which is a topic extensively covered in Cisco's BSCI exam for the CCNP and CCDP certifications.

TABLE 3-14

Addressing for 192.168.1.0/28

Network Numbers	Host Addresses	Directed Broadcast Addresses
192.168.1.0	192.168.1.1 - 192.168.1.14	192.168.1.15
192.168.1.16	192.168.1.17 - 192.168.1.30	192.168.1.31
192.168.1.32	192.168.1.33 - 192.168.1.46	192.168.1.47
192.168.1.48	192.168.1.49 - 192.168.1.62	192.168.1.63
192.168.1.64	192.168.1.65 - 192.168.1.78	192.168.1.79
192.168.1.80	192.168.1.81 - 192.168.1.94	192.168.1.95
192.168.1.96	192.168.1.97 - 192.168.1.110	192.168.1.111
192.168.1.112	192.168.1.113 - 192.168.1.126	192.168.1.127
192.168.1.128	192.168.1.129 - 192.168.1.142	192.168.1.143
192.168.1.144	192.168.1.145 - 192.168.1.158	192.168.1.159
192.168.1.160	192.168.1.161 - 192.168.1.174	192.168.1.175
192.168.1.176	192.168.1.177 - 192.168.1.190	192.168.1.191
192.168.1.192	192.168.1.193 - 192.168.1.206	192.168.1.207
192.168.1.208	192.168.1.209 - 192.168.1.222	192.168.1.123
192.168.1.224	192.168.1.225 - 192.168.1.238	192.168.1.239
192.168.1.240	192.168.1.241 - 192.168.1.254	192.168.1.255

EXERCISE 3-1

Planning IP Addressing Exercise

These last few sections dealt with how to create an addressing scheme for a network. This exercise will help reinforce these concepts, including the six steps that you should use to come up with an appropriate subnet mask value and network, directed broadcast, and host addresses.

1. You are given a Class C network (192.168.1.0) and you have 4 segments in your network, where the largest segment has 50 hosts. What subnet mask should you use and what is the layout of your addresses?

   Performing the six steps, the subnet mask is 255.255.255.192 (/26), giving you four network numbers: 192.168.1.0, 192.168.1.64, 192.168.1.128, and 192.168.1.192. Each of these four networks has a total of 64 addresses, where 62 of these can be used for host devices.

2. You are given a Class B network (172.16.0.0) and you have 490 segments in your network, where the largest segment needs 112 host addresses. What subnet mask should you use and what is the layout of your addresses?

   Performing the six steps, the subnet mask is 255.255.255.128 (/25), giving you 512 network numbers: 172.16.0.0, 172.16.0.128, 172.16.1.0, 172.16.1.128, 172.16.2.0, 172.16.2.128, and so on and so forth. Each of these 512 subnets has 128 addresses, of which 126 of these can be used to assign addressing information to host devices.

3. You are given a Class A network (10.0.0.0) and you have 9,000 segments in your network, where the largest segment needs 560 host addresses. What subnet mask should you use and what is the layout of your addresses?

   Performing the six steps, the subnet mask is 255.255.252.0 (/22), giving you 16,384 network numbers: 10.0.0.0, 10.0.4.0, 10.0.8.0, 10.0.12.0, 10.0.16.0, 10.0.20.0, 10.0.24.0, and so on and so forth. Each of these 16,384 subnets has 1,024 addresses, of which 1,022 of these can be used to assign addressing information to host devices.

exam Watch

Make sure that you practice, practice, and do more practice on exercises like the above when preparing for your exam. I can't stress this enough.

Now you should be more comfortable with planning IP addressing. In the next section, you will be presented with how to figure out whether an IP address is a network, directed broadcast, or host address.

CERTIFICATION OBJECTIVE 3.05

Figuring Out IP Address Components

For purposes of the CCNA exam, you might not be given an assignment like the one I described in the last section. However, you will have to know how to figure out how many host addresses are in a particular subnet, how many subnets you can create with a particular mask, and, given a specific IP address, is it a network, host, or directed broadcast address. The last section described how to plan IP addressing. This section, however, will teach you the tools that you will need to figure these out…more specifically, given a certain address, what type of address it is.

If you recall from the last section, there are three types of addresses for each network: a network, a directed broadcast, and host addresses. The trick to figuring this out goes back to step 4 of the last section. You need to figure out the number that networks are incrementing by. For exam purposes, you may be given an IP address and a subnet mask. Covert the decimal subnet mask to the number of bits in the mask. In our previous example, for instance 255.255.255.240 is a 28-bit mask. Take this number and subtract it from 32. In our example, this gives us 4 bits. Since the first 28 bits are network numbers, the last 4 bits are host addresses.

If you recall, every subnet has the same number of addresses. So all you need to do is raise this value to the power of 2 to figure out how many addresses are in a network, and therefore you know by how much each network number is incrementing. In our example, 2^4 gives you a total of 16 addresses in the subnet, including the network, host, and directed broadcast addresses. Based on this information, it is easy to figure out what type of address the exam is asking about.

As you will learn, subnetting is not a difficult task, but it does take **a lot of practice**. I've developed six steps to help you out. The following sections cover these six steps.

Six Step Approach For Figuring Out IP Address Components

When you are given a particular address and subnet mask, and asked whether the address is a network, host, or directed broadcast address, you should use the following six steps:

1. You need an IP address and a subnet mask (this is the easy part).
2. Examine the subnet mask and find the interesting octet. The interesting octet in the mask is the one where the network and host boundary is found. This includes the following mask values in an octet: 0, 128, 192, 224, 240, 248, 252,

and 254. It does *not* include 255--an octet with a mask value of 255 (all 8 bits are 1s) indicates that this octet is part of the network number. Only when an octet contains binary 0s does it have a host component.

3. Subtract the interesting octet in the subnet mask from 256. This will give you the increment by which network numbers are increasing in the interesting octet.
4. On a piece of paper, start writing down the network numbers, starting at with the first subnet (0), and work you way up to a network number that is higher than the address in question.
5. After you have written down the network numbers, beside each of these, write down their corresponding broadcast addresses. Remember that the broadcast address is one number less than the *next* network number. You don't have to do this with every network number…just the networks near the network number in question.
6. Between the network and broadcast addresses, right down the host addresses. Host addresses are any number between the network and directed broadcast address.

exam Watch

Remember the above six steps when trying to determine if an IP address is a network, host, or directed broadcast number.

Based on these six steps, you should then be able to figure out if your address is a host, network, or broadcast address. Note that these six steps are somewhat similar to the six steps used in the *Planning IP Addressing* section. However, the steps in this section are for test purposes and the steps in the previous section are for design purposes.

Example #1 For Figuring Out IP Address Components

In order to help you out with the six steps, let's take a look at an example as an illustration. In step 1, you have an IP address and subnet mask. Let's assume that this is **192.168.1.37 255.255.255.224** (or 192.168.1.37/27). This is a Class C network.

In step 2, you need to find the interesting octet in the subnet mask. This is the octet where the boundary exists between network and host bits. In this example, this is the fourth octet: **224**. In step 3, you need to find the increment by which network numbers are increasing. To perform this step, subtract the interesting octet from 256: **256 - 224 = 32**. Therefore, there are 32 addresses in each network, and each network is incrementing by 32 in the interesting octet (fourth octet).

exam Watch

Remember the shortcut of figuring out the multiples that network numbers are incrementing by in the interesting octet: 256 - subnet mask value = increment value.

In step 4, write down the network numbers starting with first subnet and work your way up. Here is the list of network numbers for our example: 192.168.1.0, 192.168.1.32, 192.168.1.64, 192.168.1.96, 192.168.1.128, 192.l68.1.160, 192.168.1.192, and 192.168.1.224. In this example, there are eight subnets. Mathematically, this makes sense. There are 32 addresses per subnet, with a total of 256 addresses (0-255) in a Class C network. **256 ÷ 32 = 8**! Remember that the interesting octet in the subnet mask will be the subnet number in the last subnet of the IP class address.

In step five, list the directed broadcast address beside each network number. And in step 6, list the host addresses for each network. Remember that the broadcast address for a network is one number less than the next network number and that the host addresses are any IP addresses between the network and directed broadcast addresses. Table 3-15 shows the completion of steps 5 and 6.

Given Table 3-15, the host address of 192.168.1.37 is a ***host*** address, since it falls in the rage of host addresses for subnet 192.168.1.32/27. When you are taking the CCNA exam, I wouldn't build the entire table. Instead, I would list the network numbers until you had a network number greater than the address in the question. Once this was done, for the last three network numbers, I would list the directed broadcast and host addresses, and then I would know the answer to the exam question. In the above example, these networks would be 192.168.1.0, 192.168.32.0 and 192.168.64.0.

TABLE 3-15

Network, Directed Broadcast, and Host Addresses of 192.168.1.0/27

Network Addresses	Host Addresses	Directed Broadcast Addresses
192.168.1.0	192.168.1.1 - 192.168.1.30	192.168.1.31
192.168.1.32	192.168.1.33 - 192.168.1.62	192.168.1.63
192.168.1.64	192.168.1.65 - 192.168.1.94	192.168.1.95
192.168.1.96	192.168.1.97 - 192.168.1.126	192.168.1.127
192.168.1.128	192.168.1.129 - 192.168.1.158	192.168.1.159
192.168.1.160	192.168.1.161 - 192.168.1.190	192.168.1.191
192.168.1.192	192.168.1.193 - 192.168.1.222	192.168.1.223
192.168.1.224	192.168.1.225 - 192.168.1.254	192.168.1.255

Example #2 For Figuring Out IP Address Components

Let's look at another example to help clarify the six steps. For step 1, you are given the following address and subnet mask: 192.168.1.132 255.255.255.192 (/26), which is a Class C address.

In the second step, you need to find the interesting octet in the subnet mask. This is the last octet. For a Class C network, this will *always* be the last octet. The value in this mask is **192**, indicating that the first two high-order bits in the octet are part of the network component and the last 6 low-order bits are the host component. In step 3, you need to find out by what number the network addresses are increasing by. To do this step, subtract the value in the subnet mask's interesting octet from 256: **256 - 192 = 64**. Therefore, network addresses are incrementing by 64 numbers and each network contains 64 addresses: a network address, a directed broadcast address, and 62 host addresses. Remember that since the interesting octet is in the fourth octet, the network addresses are increasing by 64 in the *interesting* (fourth) octet.

In step 4, write down the network numbers. In our example, this gives us 4 networks: 192.168.1.0, 192.168.1.64, 192.168.1.128, and 192.168.1.192. In the interesting octet, two bits are used for networking and 6 bits for host addresses. With 2 bits of networking, this gives you 4 networks: $2^2 = 4$ and with 6 bits of host addresses and 64 addresses in a network: $2^6 = 64$.

The address in question, 192.168.1.132 is between two networks: 192.168.1.128 and 192.168.1.192. This means that you should only have to perform steps 5 and 6 on these two networks, and possibly the network before it. However, let's go ahead and complete steps 5 and 6 for all of the networks since this is good practice. Remember that the directed broadcast address for a network is one number less than the next network number and that the addresses between the network and directed broadcast addresses are host addresses. Table 3-16 shows the addressing for the Class C address. Our address, 192.168.1.132 is a ***host*** address based on this table, where its network number is 192.168.1.128 and its directed broadcast is 192.168.1.191.

TABLE 3-16

Network, Directed Broadcast, and Host Addresses of 192.168.1.0/26

Network Addresses	Host Addresses	Directed Broadcast Addresses
192.168.1. 0	192.168.1.1 - 192.168.1.62	192.168.1.63
192.168.1.64	192.168.1.65 -192.168.1.126	192.168.1.127
192.168.1.128	192.168.1.129 -192.168.1.190	192.168.1.191
192.168.1.192	192.168.1.193 - 192.168.1.253	192.168.1.254

Example #3 For Figuring Out IP Address Components

The first two examples were fairly simple, since the addresses were from a Class C network. In this third example, I'll complicate matters by using a Class B network. In step 1, the address assigned is 172.16.5.0 255.255.254.0, which can also be represented as 172.16.5.0/23. This is an excellent example of an address that most test-takers would incorrectly identify on a test. Right now, I want you to guess what type of address this is (network, directed broadcast, or host) and then we'll work it through step-by-step to come up with an answer.

In step 2, you need to find the interesting octet--where the network and host boundary resides. In our case, this happens to be the *third* octet (**254**) of the subnet mask. It is important to point out that *all* of the *fourth* octet are host addresses. In step 3, you need to find the increment by which network numbers are increasing: **256 - 254 = 2**. Network numbers are incrementing by 2 in the *third* octet. This last sentence is very important. Remember that the entire fourth octet is the host component since the subnet mask value in this position is set to 0 (all 8 bits are 0).

In step 4, you need to write down your network numbers, starting with the first subnet and work your way up until you go past the IP address in question: 172.16.0.0, 172.16.2.0, 172.16.4.0, 172.16.6.0, 172.16.8.0, and so on and so forth. Remember that with a class B address, there are 16 bits in the host component. With our subnet mask, we're using 7 bits for subnets and 9 bits for hosts. Therefore, with 7 bits for subnets, we have a total of 128 subnets, where each subnet has 512 total addresses. Each network really has 510 *host* addresses, where the first and last addresses are used for the network and directed broadcast address respectively. Looking at our address, 172.16.5.0, we can tell that it at least is *not* a network address.

Let's go ahead and do steps 5 and 6, listing the directed broadcast addresses and host addresses for these subnets, as is shown in Table 3-17. Looking at this table, you can see that 172.16.5.0 is a ***host*** address! Even 172.16.0.255 is a host address! This example illustrates that you should *never* make assumptions about what type an address is without considering the subnet mask. Always remember that the subnet mask puts a context on the IP address and determines its type: network, directed broadcast, or host address.

For the CCNA exam, I would expect a trick question like this. In real life, I would typically not use addresses like 172.16.5.0 or 172.16.0.255 because this would *confuse* many network administrators. I've actually had to argue with people over the validity of these kinds of addresses as host addresses in network planning sessions. I've learned, though, it's pretty hard to teach an old dog new tricks, so instead of wasting my time arguing or explaining the address validity, I just don't use them. For test purposes, though, they *are* valid host addresses.

TABLE 3-17

Network, Directed Broadcast, and Host Addresses of 172.16.0.0/23

Network Addresses	Host Addresses	Directed Broadcast Addresses
172.16.0.0	172.16.0.1 - 172.16.1.254	172.16.1.255
172.16.2.0	172.16.2.1 - 172.16.3.254	172.16.3.255
172.16.4.0	172.16.4.1 - 172.16.5.254	172.16.5.255
172.16.6.0	172.16.6.1 - 172.16.7.254	172.16.7.255
172.16.8.0	172.16.8.1 - 172.16.9.254	172.16.9.255

EXERCISE 3-2

Determining Network, Directed Broadcast, and Host Components

These last few sections dealt with how to differentiate what type an address is: network, directed broadcast, or host address. The following exercises will help you practice your IP addressing skills.

1. You are given the following address: 192.168.1.63/255.255.255.248. What type of address is this--network, directed broadcast, or host?

   The interesting octet is the *fourth*: 248. Subtract 256 from this: 256 - 248 = 8. Network numbers are incrementing by 8: 192.168.1.0, 192.168.1.8, 192.168.1.16, 192.168.1.24, 192.168.1.32, 192.168.1.40, 192.168.1.48, 192.168.1.56, 192.168.1.64, and so on and so forth. After writing down the directed broadcast addresses, you'll see that the network 192.168.1.56 has a directed broadcast address of 192.168.1.63 and host address of 57-62. Therefore, this is a broadcast address.

2. You are given the following address: 172.16.4.255/255.255.252.0. What type of address is this--network, directed broadcast, or host?

   The interesting octet is the *third*: 252. Subtract 256 from this: 256 - 252 = 4. Network numbers are incrementing by 4 in the third octet: 172.16.0.0, 172.16.4.0, 172.16.8.0, 172.16.12.0, and so on and so forth. After writing down the directed broadcast addresses, you'll see that the network 172.16.4.0

has a directed broadcast address of 172.16.7.255 and host addresses of 172.16.4.1-172.16.7.254. Therefore, this is a host address.

CERTIFICATION SUMMARY

TCP/IP has five layers: application, transport, internet, data link, and physical. At the transport layer, TCP provides a reliable connection through the use of sequence numbers and acknowledgements. TCP uses a three-way handshake when establishing a connection: SYN, SYN/ACK, and ACK. TCP uses PAR to recover lost segments, resending segments with a delay between transmissions, until an acknowledgment is received. Applications that use TCP include FTP (21), HTTP (80), SMTP (25), and telnet (23). UDP provides unreliable connections and is more efficient that TCP. Examples of applications that use UDP include DNS (53), RIP (520), SNMP (161) and TFTP (69).

IP functions at the internet layer and includes protocols like ICMP, ARP, RARP, OSPF, and others. ICMP is used to test connections. Ping uses ICMP echo messages to test layer-3 connectivity. ARP resolves an IP address to a MAC address. RARP, used by BOOTP and DHCP, resolve a MAC address to an IP address (used to acquire IP addressing information on a device). IP addresses are 32-bits in length and are broken up into 4 bytes, with a period between the bytes. This format is referred to as dotted decimal.

There are five classes of IP addresses: A (1-126), B (128-191), C (192-223), D (224-239), and E (240-254). Class A addresses have one network byte and three host bytes. Class B addresses have two network and two host bytes. Class C addresses have three network bytes and one host byte. Private IP addresses include networks 10.0.0.0/8, 172.16.0.0/16-172.31.0.0/16, and 192.168.0.0/24-192.168.255.0/24.IP addresses have three components: network, host, and broadcast. The very first number in a network is the network, or wire number. The very last address is the broadcast address of the network. Any addresses between the network and broadcast addresses are host addresses. What differentiates a network, host, and broadcast address is the context the subnet mask places on the address. The subnet mask is used to mark the boundary between the network and host bits.

TWO-MINUTE DRILL

TCP/IP Protocol Stack

- ❑ The TCP/IP protocol stack has the following layers: Physical, Data Link, Internet, Transport, and Application.
- ❑ The transport layer provides flow control (through the use of windowing), reliable connections, (through the user of sequence numbers and acknowledgments), and multiplexing (allowing multiples applications to simultaneously send and receive data). TCP provides reliable connections and goes through a 3-way handshake to establish a connection whereas UDP provides unreliable connections.
- ❑ The Internet layer corresponds to the Network layer of the OSI Reference Model. Many protocols function at this layer, like ARP, RARP, and ICMP. ARP resolves IP to MAC addresses, RARP is used by BOOTP and DHCP to help a device acquire an IP address, and ICMP is used to send error and control information. The ping utility uses ICMP.

IP Addressing Introduction

- ❑ IP addresses are 32 bits in length, and are broken into four bytes (8 bits) with a period between the bytes. This format is called dotted decimal.
- ❑ Hexadecimal digits are represented in 4 bit values, ranging from 0-F.
- ❑ IP addresses are broken into five classes: A (1-126), B (128-191), C (192-223), D (224-239) and E (240-254). IP addresses are broken into two components: network and host. With Class A addresses, the first byte is a network number, Class B, the first two bytes, and Class C, the first three bytes.
- ❑ The first few bits in the first octet identity the class of address. Class A addresses begin with "0", Class B with "10", Class C with "110", Class D with "1110" and Class E with "11110".
- ❑ Each network has three components to its address: network, directed broadcast, and host. The first number in the network is the network address, the last is the directed broadcast address, and any addresses between these two are host addresses.

Subnetting

- Subnetting allows you to break up and use your addressing space more efficiently. Basically, subnetting steals the higher-order bit or bits from the host component and uses these bits to create more subnets, with a smaller number of host addresses.
- Subnet masks are 32 bits long and are typically represented in dotted decimal, like 255.255.255.0 or the number of networking bits, like "/24". The networking bits in a mask must be contiguous and the host bits in the subnet mask must be contiguous. 255.0.255.0 is an invalid mask.

Planning IP Addressing

- There are six steps to design a network with IP addresses: 1) figure out your network and host requirements; 2) satisfy host and network requirements; 3) figure out the subnet mask; 4) figure out the network addresses; 5) figure out the directed broadcast addresses; 6) figure out the host addresses
- When satisfying your host and networking requirements, you need to figure out how many bits you need to in order to meet your network segment requirements and how many bits you need in order to satisfy the maximum number of hosts on the largest segment in your network. When you add these two values together, they shouldn't exceed the original number of host bits in the host component of the address.

Figuring Out IP Address Components

- Use six steps in order to figure out the type of address: 1) List the IP address and mask; 2) find the interesting octet in the subnet mask; 3) Subtract the interesting octet from 256, which gives you the increment that network addresses are increasing by in the interesting octet; 4) Write down the network addresses; 5) Beside each network address, write down its directed broadcast address; 6) Host addresses are addresses between the network and directed broadcast addresses.
- When figuring out directed broadcast addresses, they will be one number less than the next network address.
- Subnet masks determine the context of IP addresses…whether an address is a network, broadcast, or host address.

SELF TEST

The following Self Test questions will help you measure your understanding of the material presented in this chapter. Read all the choices carefully, as there may be more than one correct answer. Choose all correct answers for each question.

TCP/IP Protocol Stack

1. The TCP/IP protocol stack has _____ layers.
   - A. 4
   - B. 5
   - C. 6
   - D. 7
2. Which of the following is not true concerning TCP?
   - A. Provides for reliable connections
   - B. Uses windowing for flow control
   - C. Multiplexes applications
   - D. Is more efficient than UDP
3. Which of the following is a Network layer protocol for the TCP/IP protocol stack?
   - A. TCP
   - B. UDP
   - C. ICMP
   - D. None of these

IP Addressing Introduction

4. You have this binary value: 11000001. This equates to _____ in decimal.
5. A Class A address has _____ host bits.
   - A. 8
   - B. 16
   - C. 20
   - D. 24

6. 191.75.39.24 is a Class _____ address.
   A. A
   B. B
   C. C
   D. None of the above
7. 172.16.240.256 is a class _____ address.
   A. A
   B. B
   C. C
   D. None of the above

Subnetting

8. Which of the following is a valid subnet mask value?
   A. 255.0.255.255
   B. 0.0.0.255
   C. 255.255.254.0
   D. 255.255.255.256
9. The function of a _____ is to differentiate between the network address, the host addresses, and the directed broadcast address.

Planning IP Addressing

10. You are given a Class C network with 25 bits of networking. How many subnets do you have?
    A. 1
    B. 2
    C. 3
    D. 4
11. You are given a Class C network with a subnet mask of 255.255.255.248. How many host addresses are there on each subnet?
    A. 4
    B. 6
    C. 8
    D. 14

12. You are given a Class B network with a subnet mask of 255.255.255.192. How many host addresses are there on each subnet?

A. 30
B. 62
C. 126
D. 254

Figure Out IP Address Components

13. You are given the following addressing information: 192.168.37.192/25. What type of address is this?

A. Network
B. Directed Broadcast
C. Host

14. You are given the following addressing information: 172.17.16.255/23. What type of address is this?

A. Network
B. Directed Broadcast
C. Host

15. You are given the following addressing information: 10.0.8.0/22. What type of address is this?

A. Network
B. Directed Broadcast
C. Host

16. You are given a MAC address of 01A2.0482.FE12. What is the OUI value in binary?

A. 00011011.10100010.00000100
B. 00000000.00000001.00000100
C. 00000001.10100010.00000100
D. 00000001.10100001.00000010

SELF TEST ANSWERS

TCP/IP Protocol Stack

1. ☑ **B.** The TCP/IP protocol stack has 5 layers: Physical, Data Link, Internet, Transport, and Application.
   ☒ **A**, **C**, and **D** are incorrect.
2. ☑ **D.** UDP is more efficient than TCP since it doesn't have the overhead associated with it that TCP has. For instance, UDP doesn't go through a 3-way handshake to set up a connection and doesn't use sequence and acknowledgment numbers to implement flow control.
   ☒ **A**, **B**, and **C** are all true concerning TCP: it provides reliable connections, windowing for flow control, and multiplexing for applications.
3. ☑ **D.** The TCP/IP protocol stack doesn't have a Network layer…it has an Internet layer.
   ☒ **A** and **B** are transport layer protocols. **C** is an Internet layer protocol.

IP Addressing Introduction

4. ☑ 193.
5. ☑ **D.** Class A addresses have 24 host bits and 8 networking bits.
   ☒ **A** is true for Class C networks. **B** is true for Class B networks. **C** can only be true for subnetted Class A and B networks.
6. ☑ **B.** 191.75.39.24 is a Class B network. Class B networks range from 128-191.
   ☒ **A** addresses range from 1-126. **C** addresses range from 192-223. Since there is an answer, **D** is incorrect.
7. ☑ **D.** It's impossible to represent 256 in a byte--the values range from 0-255.
   ☒ **A**, **B**, and **C** are incorrect.

Subnetting

8. ☑ **C.** 255.255.254.0 is a valid subnet mask--the 1s and 0s must be contiguous.
   ☒ **A** has noncontiguous 1s. **B** is an inverted mask, with the network and host bits reversed. **D** has an invalid mask value in the fourth octet: 256.
9. ☑ **Subnet mask**. The function of a subnet mask is to differentiate between the network address, the host addresses, and the directed broadcast address.

Planning IP Addressing

10. ☑ **B.** Class C networks have 24 bits--this example steals on bit. 2 raised to the power of 1 equals 2 subnets.
☒ **A**, **C**, and **D** are incorrect.

11. ☑ **B.** There are 3 host bits, with 2 raised to the power of 3 resulting in 8 addresses in a network, but you lose 2 for the network and directed broadcast address, resulting in 6 host addresses. You could also subtract 248 from 256, resulting in a total of 8 addresses per network, of which the first and last are reserved.
☒**A**, **C**, and **D** are incorrect.

12. ☑ **B.** There are 6 host bits, with 2 raised to the power of 6 resulting in 64 addresses in a network, but you lose 2 for the network and directed broadcast address, resulting in 62 host addresses. You could also subtract 192 from 256, resulting in a total of 64 addresses in a network…but you can't use the first and the last.
☒ **A**, **C**, and **D** are incorrect.

Figure Out IP Address Components

13. ☑ **C.** There is 1 subnet bit for this Class C network, resulting in two networks: 192.168.37.0 and 192.168.37.128, making 192.168.37.192 a host address. Host addresses for this subnet range from 192.168.37.129-192.168.37.254.
☒ **A** is true for 192.168.37.0 and 192.168.37.128. **B** is true for 192.168.37.127 and 192.168.37.255.

14. ☑ **C.** Network addresses are incrementing by 2 in the *third* octet. Where 172.17.16.255 is a host address. Host addresses range from 172.17.16.1-172.17.17.254.
☒ **A** is true for 172.17.16.0. **B** is true for 172.17.17.255.

15. ☑ **A.** Network addresses are incrementing by 4 in the third octet. Where 10.0.8.0 is a network address. Other network addresses include 10.0.0.0, 10.0.4.0, 10.0.8.0, 10.0.12.0, and so on and so forth.
☒ **B** is true for 10.0.3.255, 10.0.7.255, 10.0.11.255, 10.0.15.255, and so on and so forth. **C** is true for 10.0.8.1-10.0.11.255, as well as other host addresses in other subnets.

16. ☑ **C.** Remember that the OUI portion is the first six hexadecimal characters in a MAC address (Chapter 2). Hex characters are represented in 4 bits in binary. 0=0000, 1=0001, A=1010, 2=0010, 0=0000, and 4=0100. When you concatenate these together, the result is 00000001.01000010.0100 (the first six hexadecimal digits).
☒ **A** converts to 0B for the first byte, making it incorrect. **B** converts to 00 for the first byte, making it incorrect. D's first byte is correct, but the second byte converts to A1 and third byte to 02, making it incorrect