chapter 7

TCP/IP Basics

"If it's sent by ship then it's a cargo, if it's sent by road then it's a shipment."

—Dave Allen

In this chapter, you will learn how to

- **Describe how the Internet Protocol works**
- **Explain CIDR and subnetting**
- **Describe the functions of static and dynamic IP addresses**

The mythical MHTechEd network (remember that from Chapter 2?) provided an overview of how networks work. At the bottom of every network, at OSI Layers 1 and 2, resides the network hardware: the wires, switches, network cards, and more that enable data to move physically from one computer to another. Above the Physical and Data Link layers, the "higher" layers of the model—network protocols and applications—work with the hardware to make the network magic happen.

Chapters 3 through 6 provided details of the hardware at the Physical and Data Link layers. You learned about the network protocols, such as Ethernet, that create uniformity within networks, so that the data frame created by one NIC can be read properly by another NIC.

This chapter begins a fun journey into the software side of networking. You'll learn the details about the IP addressing scheme that enables computers in one network to communicate with each other and computers in other networks. You'll get the full story on how TCP/IP networks divide into smaller units—subnets—to make management of a large TCP/IP network easier. And you won't just get it from a conceptual standpoint. This chapter provides the details you've undoubtedly been craving, teaching you how to set up a network properly. The chapter finishes with an in-depth discussion on implementing IP addresses.

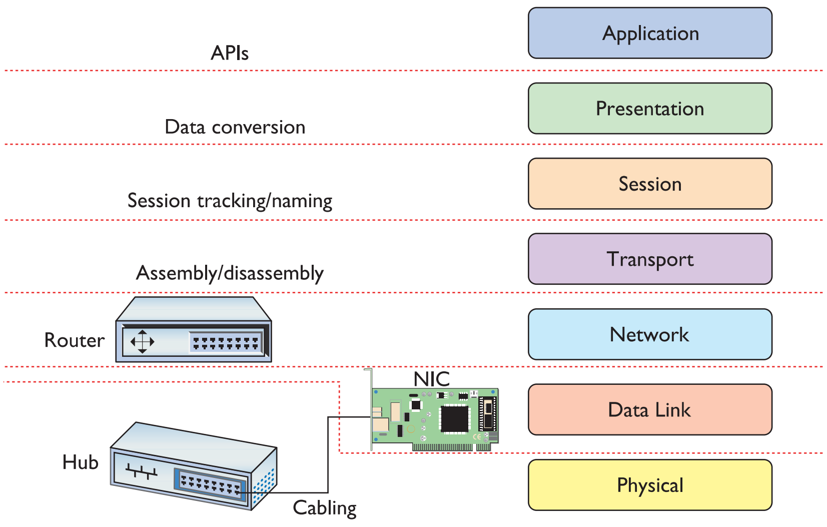
Historical/Conceptual

The early days of networking software saw several competing standards that did not work well together. Novell NetWare, Microsoft Windows, and Apple Macintosh ran networking software to share folders and printers, while the UNIX/Linux world did crazy things like sharing terminals—handy for the UNIX/Linux users, but it made no sense to the Windows folks—and there was this new thing called e-mail (like that was ever going to go anywhere). The Internet had just been opened to the public. The World Wide Web was just a plaything for programmers and scientists. All of these folks made their own software, interpreting (or totally ignoring) the OSI model the way they wanted to, and all trying (arguably) to become THE WAY the whole world was going to network. It was an unpleasant, ugly world for guys like me who had the audacity to try to make, for example, a UNIX box work with a Windows computer.

The problem was that no one agreed on how a network should run. Everyone's software had its own set of Rules of What a Network Should Do and How to Do It. These sets of rules—and the software written to follow these rules—were broken down into individual rules called **protocols**. Each set of rules had many protocols lumped together under the term **protocol suite**. Novell NetWare called its protocol suite IPX/SPX, Microsoft called its NetBIOS/NetBEUI, Apple called its AppleTalk, and the UNIX folks used this wacky protocol suite called TCP/IP.

Well, TCP/IP has won. Sure, you may find the occasional network still running one of these other protocol suites, but they're rare these days. To get ahead in today's world, to get on the Internet, and to pass the CompTIA Network+ exam, you only need to worry about TCP/IP. Novell, Microsoft, and Apple no longer actively support anything but TCP/IP. You live in a one-protocol-suite world, the old stuff is forgotten, and you kids don't know how good you got it!

TCP/IP fits nicely into the OSI seven-layer model, occupying Layers 3–5: Network, Transport, and Session, respectively (Figure 7.1). Starting from the bottom up, the **Internet Protocol (IP)**—both version 4 (IPv4), which interests us here, and version 6 (IPv6), which is covered in the chapter of the same name, Chapter 13—stands by itself at Layer 3. IP handles all of the logical addressing issues. The Transmission Control Protocol (TCP), the User Datagram Protocol (UDP), and the Internet Control Message Protocol (ICMP) operate at the Transport and Session layers. These protocols define how the connections take place between two computers. All of these individual protocols work together to make the TCP/IP protocol suite.



● **Figure 7.1** OSI redux

Test Specific

IP in Depth

TCP/IP supports simple networks and complex networks. You can use the protocol suite to connect a handful of computers to a switch and create a local area network (LAN). TCP/IP also enables you to interconnect multiple LANs into a wide area network (WAN).

At the LAN level, all the computers use Ethernet, and this creates a hurdle for WAN-wide communication. For one computer to send a frame to another computer, the sending computer must know the MAC address of the destination computer. This begs the question: How does the sender get the recipient's MAC address?

Cross Check

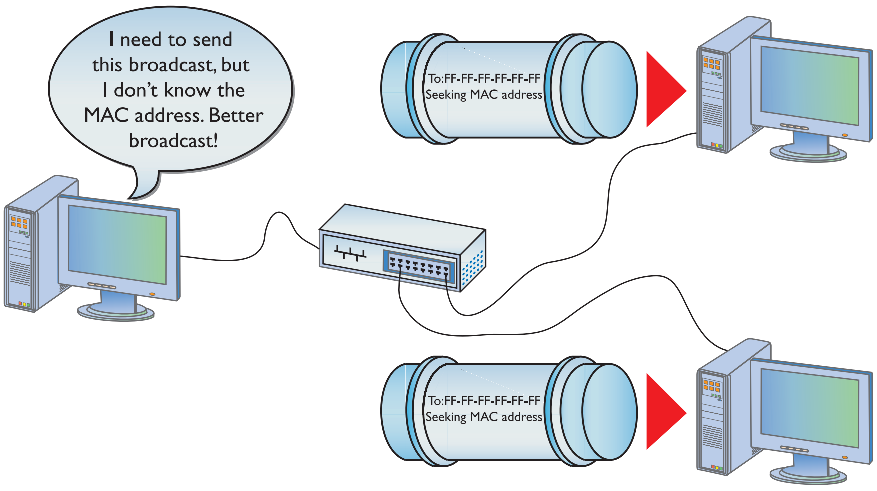
Broadcasting

You first ran into broadcasting way back in Chapter 2, "Building a Network with the OSI Model," so check your memory now. What happens to the broadcast frame? Does it reach all the computers on a LAN? How many computers actually process that broadcast frame?

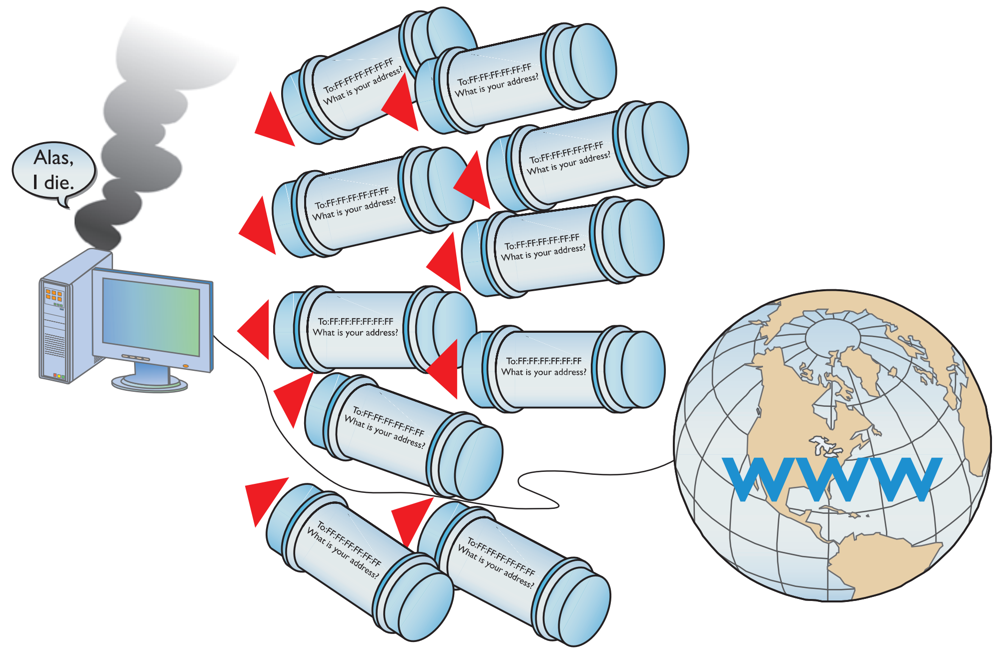
In a small network this is easy. The sending computer simply *broadcasts* by sending a frame to MAC address FF-FF-FF-FF-FF-FF, the universal MAC address for broadcast. Figure 7.2 shows a computer broadcasting for another computer's MAC address.

Broadcasting takes up some of the network bandwidth, but in a small network it's acceptably small. But what would happen if the entire Internet used broadcasting (Figure 7.3)? In that case the whole Internet would come to a stop.

TCP/IP networks use **IP addressing** to overcome the limitations inherent in Ethernet networks. IP addresses provide several things. First, every machine on a TCP/IP network—small or large—gets a unique IP address that identifies the machine. Second, IP addresses group together sets of computers into logical networks, so you can distinguish one LAN from another, for example. Finally, because TCP/IP network equipment understands the IP


I need to send this broadcast, but I don't know the MAC address. Better broadcast!
To:FF-FF-FF-FF-FF-FF Seeking MAC address
To:FF-FF-FF-FF-FF-FF Seeking MAC address

• Figure 7.2 PC broadcasting for a MAC address



• **Figure 7.3** Broadcasting won't work for the entire Internet.

addressing scheme, computers can communicate with each other *between* LANs, in a WAN, without broadcasting for MAC addresses.

Chapter 2 touched on IP addresses briefly, but network techs need to understand them intimately. Let's dive into the structure and function of the IP addressing scheme.

IP Addresses

The most common type of IP address (officially called IPv4, but usually simplified to just "IP") consists of a 32-bit value. Here's an example of an IP address:

11000000101010000000010000000010

Whoa! IP addresses are just a string of 32 binary digits? Yes they are, but to make IP addresses easier to use for us humans, the 32-bit binary value is broken down into four groups of eight, separated by periods or *dots* like this:

11000000.10101000.00000100.00000010

Each of these 8-bit values is in turn converted into a decimal number between 0 and 255. If you took every possible combination of eight binary values and placed them in a spreadsheet it would look something like the list in the left column. The right column shows the same list with a decimal value assigned to each.

00000000	00000000 = 0
00000001	00000001 = 1
00000010	00000010 = 2

00000011	00000011 = 3
00000100	00000100 = 4
00000101	00000101 = 5
00000110	00000110 = 6
00000111	00000111 = 7
00001000	00001000 = 8
(skip a bunch in the middle)	*(skip a bunch in the middle)*
11111000	11111000 = 248
11111001	11111001 = 249
11111010	11111010 = 250
11111011	11111011 = 251
11111100	11111100 = 252
11111101	11111101 = 253
11111110	11111110 = 254
11111111	11111111 = 255

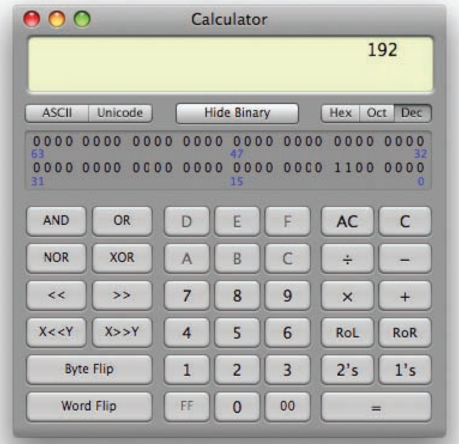
When you type an IP address into a computer, the periods are ignored and the decimal numbers are immediately converted into binary. People need dotted decimal, the computers do not.

Using a calculator utility to convert to and from binary/decimal is a critical skill for a network tech. Later on you'll do this again, but by hand!

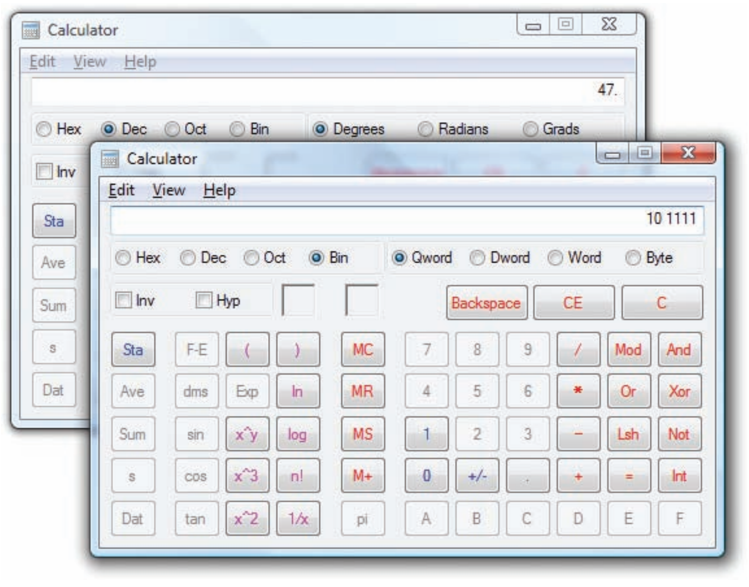
Converted, the original value of 11000000.10101000.00000100.00000010 is displayed as 192.168.4.2, IPv4's **dotted decimal notation** (also referred to as the *dotted-octet numbering system*). Note that dotted decimal is simply a shorthand way for people to discuss and configure the binary IP addresses computers use.

People who work on TCP/IP networks must know how to convert dotted decimal to binary and back. It's easy to convert using any operating system's calculator. Every OS has a calculator (Linux/UNIX systems have about 100 different ones to choose from) that has a scientific or programmer mode like the one shown in Figure 7.4.

To convert from decimal to binary, just go into decimal view, type in the value, and then switch to binary view to get the result. To convert to decimal, just go into binary view, enter the binary value, and switch to decimal view to get the result. Figure 7.5 shows the results of Windows Vista's Calculator converting the decimal value 47 into binary. Notice the result is



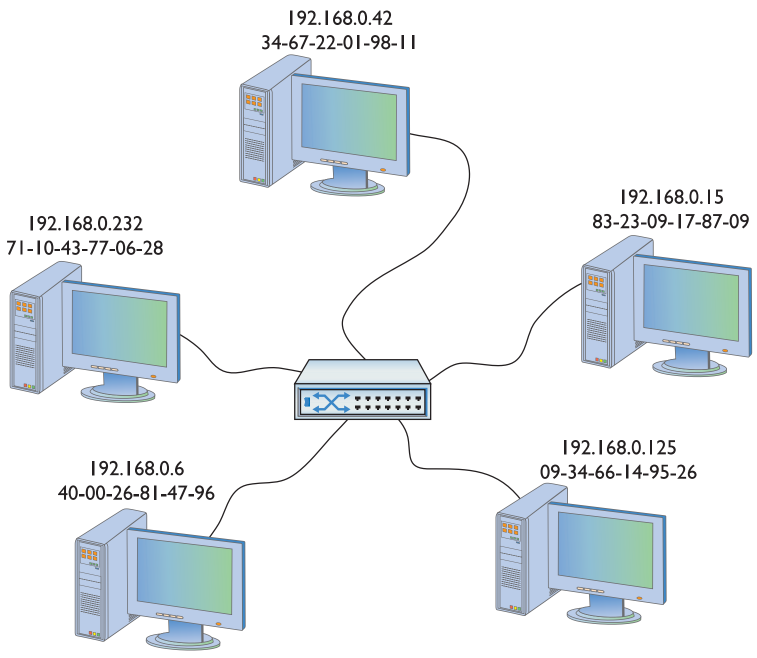
• **Figure 7.4** Macintosh OS X Calculator in Programmer mode


Calculator
Edit View Help
47.
Hex Dec Oct Bin Degrees Radians Grads
Inv
Sta
Ave
Sum
s
Dat
Calculator
Edit View Help
10 1111
Hex Dec Oct Bin Qword Dword Word Byte
Inv Hyp
Backspace CE C
Sta F-E ( ) MC 7 8 9 / Mod And
Ave dms Exp ln MR 4 5 6 * Or Xor
Sum sin x^y log MS 1 2 3 - Lsh Not
s cos x^3 n! M+ 0 +/- . + = Int
Dat tan x^2 1/x pi A B C D E F

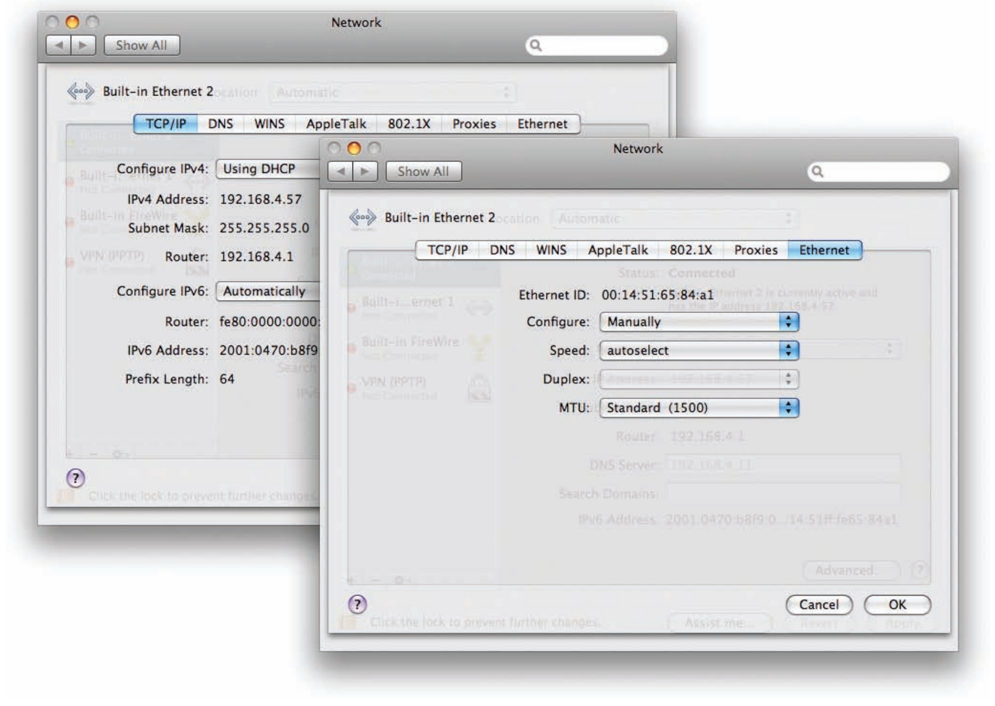
• Figure 7.5 Converting decimal to binary with Windows Vista's Calculator

101111—the leading two zeroes do not appear. When you work with IP addresses you must always have eight digits, so just add two more to the left to get 00101111.

Just as every MAC address is unique on a network, every IP address must be unique as well. For logical addressing to work, no two computers on the same network may have the same IP address. In a small network running TCP/IP, every computer has both an IP address and a MAC address (Figure 7.6).



• Figure 7.6 Small network with both IP and MAC addresses



Configure IPv4: Using DHCP
IPv4 Address: 192.168.4.57
Subnet Mask: 255.255.255.0
Router: 192.168.4.1
Configure IPv6: Automatically
Router: fe80:0000:0000:
IPv6 Address: 2001:0470:b8f9
Prefix Length: 64

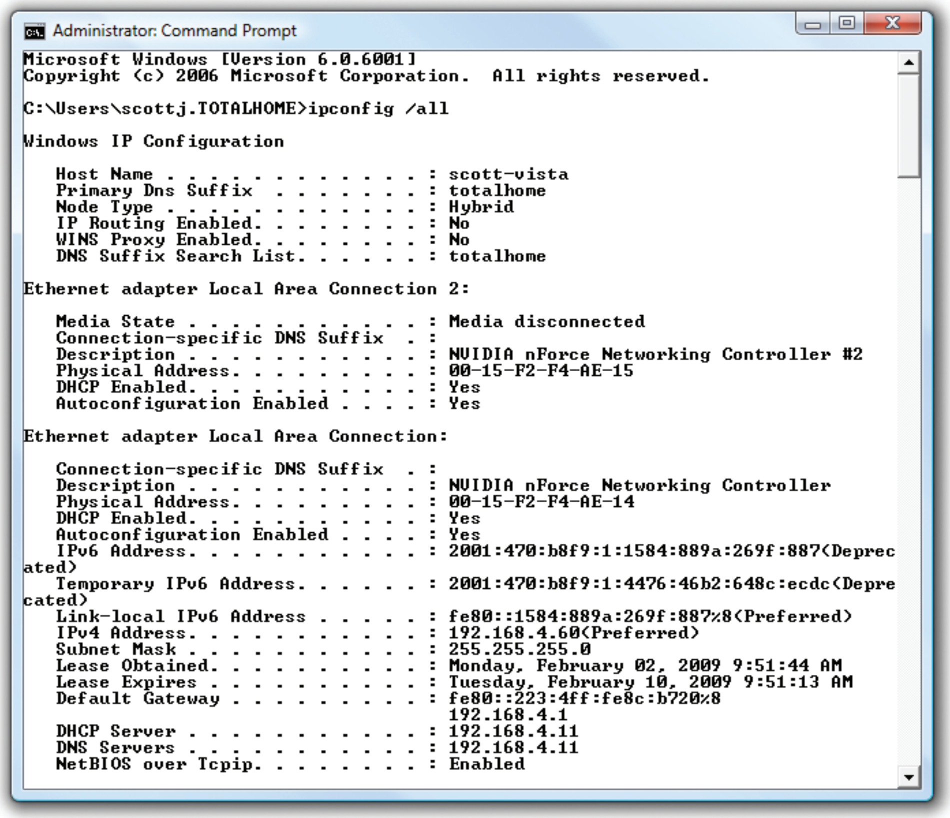
Ethernet ID: 00:14:51:65:84:a1
Configure: Manually
Speed: autoselect
Duplex:
MTU: Standard (1500)

• **Figure 7.7** Macintosh OS X Network utility

Every operating system comes with a utility (usually more than one utility) to display a system's IP address and MAC address. Figure 7.7 shows a Macintosh OS X system's Network utility. Note the MAC address (00:14:51:65:84:a1) and the IP address (192.168.4.57).

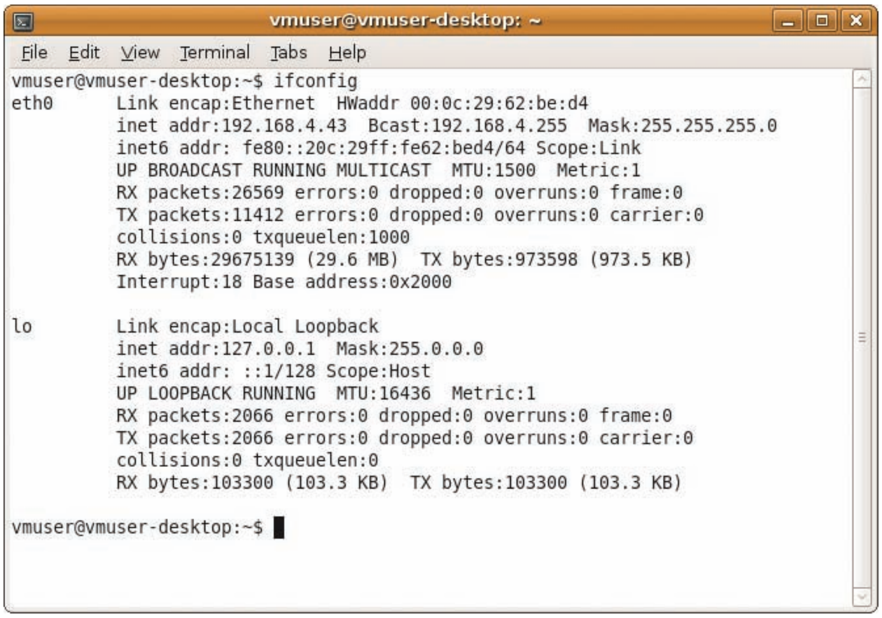
Every operating system also has a command-line utility to give you this information. In Windows, for example, you can use **IPCONFIG** to display the IP and MAC addresses. Run `ipconfig /all` to see the results shown in Figure 7.8.

In the UNIX/Linux/Mac OS X world, you can run the very similar



```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Users\scottj.TOTALHOME>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : scott-vista
   Primary Dns Suffix  . . . . . . . : totalhome
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : totalhome

Ethernet adapter Local Area Connection 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : NVIDIA nForce Networking Controller #2
   Physical Address. . . . . . . . . : 00-15-F2-F4-AE-15
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : NVIDIA nForce Networking Controller
   Physical Address. . . . . . . . . : 00-15-F2-F4-AE-14
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv6 Address. . . . . . . . . . . : 2001:470:b8f9:1:1584:889a:269f:887(Deprec
ated)
   Temporary IPv6 Address. . . . . . : 2001:470:b8f9:1:4476:46b2:648c:ecdc(Depre
cated)
   Link-local IPv6 Address . . . . . : fe80::1584:889a:269f:887%8(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.4.60(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Monday, February 02, 2009 9:51:44 AM
   Lease Expires . . . . . . . . . . : Tuesday, February 10, 2009 9:51:13 AM
   Default Gateway . . . . . . . . . : fe80::223:4ff:fe8c:b720%8
                                       192.168.4.1
   DHCP Server . . . . . . . . . . . : 192.168.4.11
   DNS Servers . . . . . . . . . . . : 192.168.4.11
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

• **Figure 7.8** IPCONFIG /ALL



```
vmuser@vmuser-desktop:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:62:be:d4
          inet addr:192.168.4.43  Bcast:192.168.4.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe62:bed4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:26569 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11412 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:29675139 (29.6 MB)  TX bytes:973598 (973.5 KB)
          Interrupt:18 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2066 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2066 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:103300 (103.3 KB)  TX bytes:103300 (103.3 KB)

vmuser@vmuser-desktop:~$ █
```

• **Figure 7.9** IFCONFIG in Ubuntu

Make sure you know that IPCONFIG and IFCONFIG provide a tremendous amount of information regarding a system's TCP/IP settings.

IFCONFIG command. Figure 7.9, for example, shows the result of an IFCONFIG ("eth0" is the NIC).

IP Addresses in Action

IP addresses support both LANs and WANs. This can create problems in some circumstances, such as when a computer needs to send data both to computers in its own network and to computers in other networks. How can this be accomplished?

To make all this work, IP must do three things:

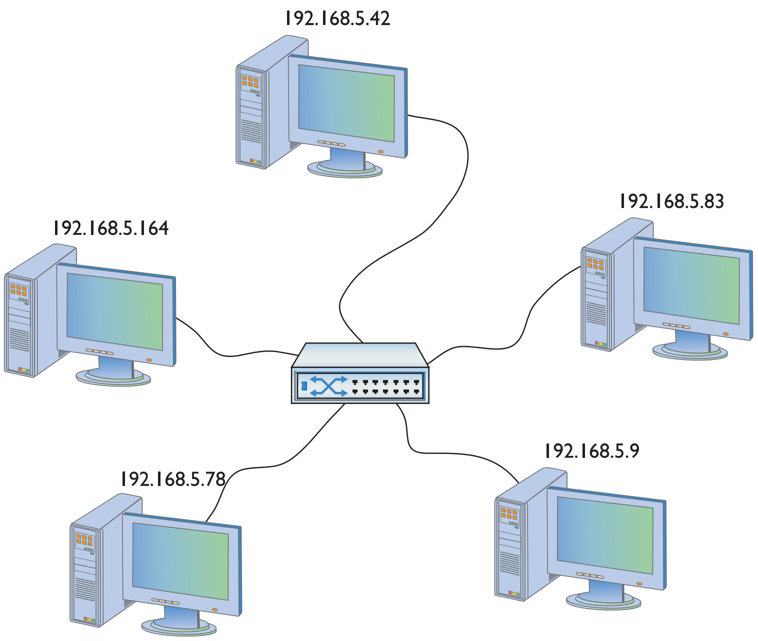
1. Create some way to use IP addresses such that each LAN has its own identification.
2. Interconnect all of the LANs together using routers and give those routers some way to use the network identification to send packets to the right network.
3. Give each computer on the network some way to recognize if a packet is for the LAN or for a computer on the WAN so it knows how to handle the packet.

Network IDs

The network ID and the host ID are combined to make a system's IP address.

To differentiate LANs from one another, each computer on a single LAN must share a very similar IP address where some of the IP address—reading left to right—matches all the others on the LAN. Figure 7.10 shows a LAN where all of the computers share the first three numbers of the IP address, with only the last number being unique on each system.

In this example, every computer has an IP address of 192.168.5.*x*. That means the **network ID** is 192.168.5.0. The *x* part of the IP address is the **host ID**. Combine the network ID (after dropping the ending 0) with the host ID to get an individual system's IP address. No individual computer can have an IP address that ends with 0 because that is reserved for network IDs.

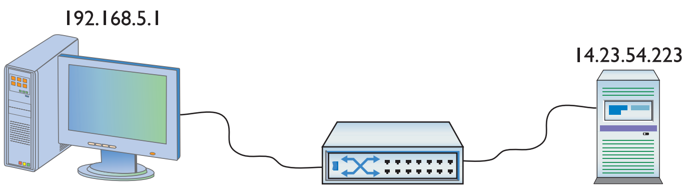


• Figure 7.10 IP addresses for a LAN

Interconnecting

To organize all those individual LANs into a larger network, every TCP/IP LAN that wants to connect to another TCP/IP LAN must have a router connection. There is no exception to this critical rule. A router therefore needs an IP address on the LANs that it serves (Figure 7.11) so that it can correctly route packets.

The IP address of the router's connection to your LAN is known as the **default gateway**. Most network administrators give the LAN-side NIC on the default gateway the lowest host address in the network, usually the host ID of 1.



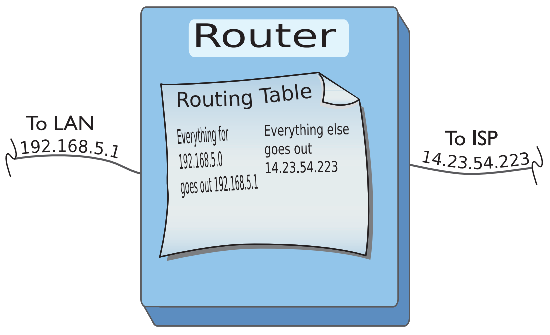
• Figure 7.11 LAN with router

Routers use network IDs to determine network traffic. Figure 7.12 shows a diagram for a small, two-NIC router similar to the ones you'd see in many homes. Note that one port (192.168.5.1) connects to the LAN and the other port connects to the Internet service provider's network (14.23.54.223). Built into this router is a **routing table**, the actual instructions that tell the router what to do with incoming packets and where to send them.

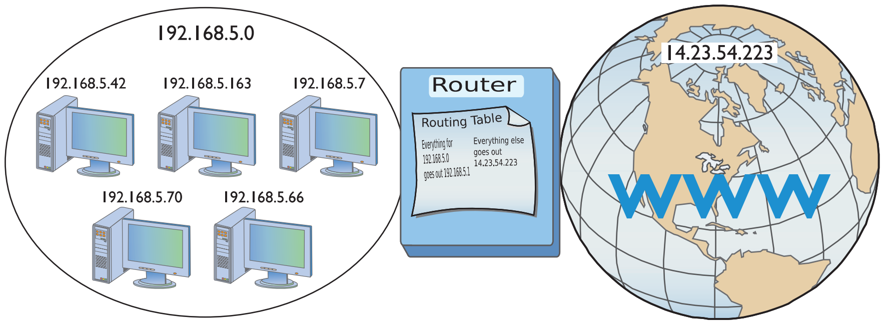
Routing tables are covered in more detail in Chapter 8.

Now let's add in the LAN and the Internet (Figure 7.13). When discussing networks in terms of network IDs, by the way, especially with illustrations in books, it's common practice to draw circles around stylized networks. It's the IDs you should concentrate on here, not the specifics of the networks.

Network IDs are very flexible, as long as no two interconnected networks share the same network ID. If you wished, you could change the network ID of the 192.168.5.0 network to 192.155.5.0, or 202.21.8.0, just as


Router
Routing Table
Everything for 192.168.5.0 goes out 192.168.5.1
Everything else goes out 14.23.54.223
To LAN
192.168.5.1
To ISP
14.23.54.223

• Figure 7.12 Router diagram



• **Figure 7.13** LAN, router, and the Internet

long as you guarantee no other LAN on the WAN shares the same network ID. On the Internet, powerful governing bodies make sure no two LANs share the same network ID by carefully allocating the network IDs. I'll talk more about how this works later on in the chapter.

So far you've only seen examples of network IDs where the last value is zero. This is common for small networks, but it creates a limitation. With a network ID of 192.168.5.0, for example, a network is limited to IP addresses from 192.168.5.1 to 192.168.5.254. (192.168.5.255 is a broadcast address used to talk to every computer on the LAN.) This provides only 254 IP addresses: enough for a small network, but many organizations need many more IP addresses. No worries! It's easy enough to simply make a network ID with more zeroes, such as 170.45.0.0 (for a total of 65,534 hosts) or even 12.0.0.0 (for around 16.7 million hosts).

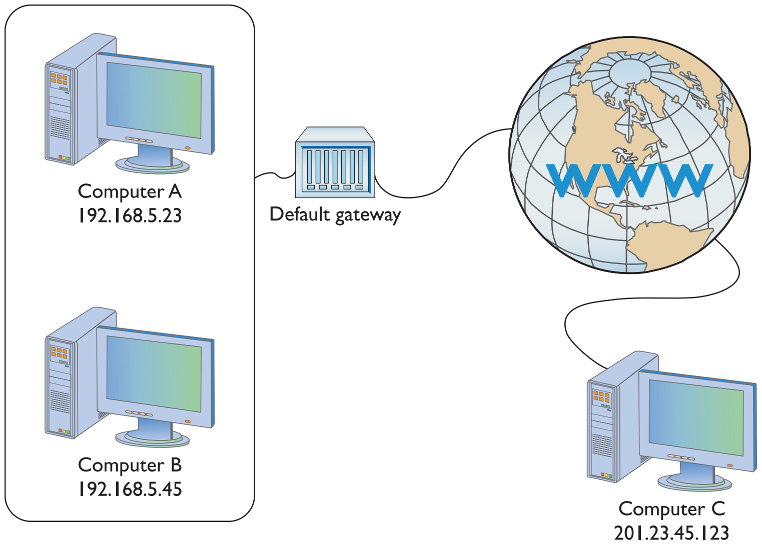
Network IDs enable you to connect multiple LANs into a WAN. Routers connect everything together and use routing tables to keep track of which packets go where. So that handles task number two.

Now that you've seen how IP addressing works with LANs and WANs, let's turn to how IP establishes a way for each computer on the network to recognize if a packet is to a computer on the LAN or to a computer on the WAN so it knows how to handle the packet. The secret to this is something called the subnet mask.

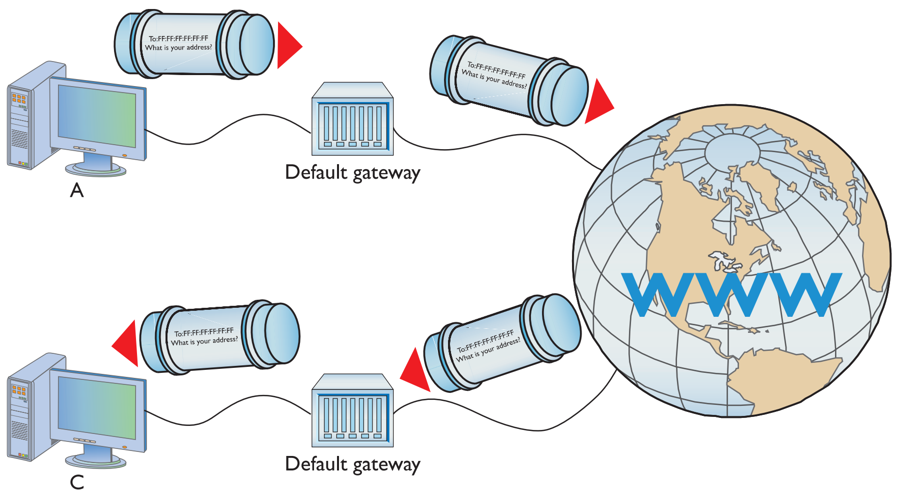
Subnet Mask

Picture this scenario. Three friends sit at their computers—Computers A, B, and C—and want to communicate with each other. Figure 7.14 illustrates the situation. You can tell from the drawing that Computers A and B are in the same LAN, whereas Computer C is on a completely different LAN. The IP addressing scheme can handle this communication, so let's see how it works.

The process to get a packet to a local computer is very different from the process to get a packet to a faraway computer. If one computer



• **Figure 7.14** The three amigos, separated by walls or miles



• **Figure 7.15** Sending a packet remotely

wants to send a packet to a local computer, it must send out a broadcast to get the other computer's MAC address, as you'll recall from earlier in the chapter and Figure 7.2. (It's easy to forget about the MAC address, but remember that the network uses Ethernet and *must* have the MAC address to get the packet to the other computer.) If the packet is for some computer on a faraway network, the sending computer must send the packet to the default gateway (Figure 7.15).

In the scenario illustrated in Figure 7.14, Computer A wants to send a packet to Computer B. Computer B is on the same LAN as Computer A, but that begs a question: How does Computer A know this? Every TCP/IP computer needs a tool to tell the sending computer whether the destination IP address is local or long distance. This tool is the subnet mask.

A **subnet mask** is nothing more than a string of ones followed by some number of zeroes, always totaling exactly 32 bits, typed into every TCP/IP host. Here's an example of a typical subnet mask:

11111111111111111111111100000000

For the courtesy of the humans reading this (if there are any computers reading this book, please call me—I'd love to meet you!) let's convert this to dotted decimal. First add some periods:

11111111.11111111.11111111.00000000

Then convert each octet into decimal (use a calculator):

255.255.255.0

When you line an IP address up with a corresponding subnet mask in binary, the portion of the IP address that aligns with the ones of the subnet mask is the network ID portion of the IP address. The portion that aligns with the zeroes is the host ID. With simple IP addresses, you can see this with dotted decimal, but you'll want to see this in binary for a true understanding of how the computers work.

The IP address 192.168.5.23 has a subnet mask of 255.255.255.0. Convert both numbers to binary and then compare the full IP address to the ones and zeroes of the subnet mask:

	Dotted Decimal	**Binary**
IP address	192.168.5.23	11000000.10101000.00000101.00010111
Subnet mask	255.255.255.0	11111111.11111111.11111111.00000000
Network ID	192.168.5.0	11000000.10101000.00000101.*x*
Host ID	*x.x.x*.23	*x.x.x*.00010111

Before a computer sends out any data, it first compares the destination IP address to its own IP address using the subnet mask. If the destination IP address matches the computer's IP wherever there's a 1 in the subnet mask, then the sending computer knows it's a local destination. The network IDs match. If even 1 bit of the destination IP address where the ones are on the subnet mask is different, then the sending computer knows it's a long-distance call. The network IDs do not match.

At this point you should memorize that 0 = 00000000 and 255 = 11111111. You'll find this very helpful throughout the rest of the book.

Let's head over to Computer A and see how the subnet mask works. Computer A's IP address is 192.168.5.23. Convert that into binary:

11000000.10101000.00000101.00010111

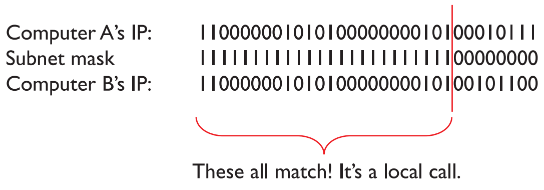
Now drop the periods, because they mean nothing to the computer:

11000000101010000000010100010111

Let's say Computer A wants to send a packet to Computer B. Computer A's subnet mask is 255.255.255.0. Computer B's IP address is 192.168.5.45. Convert this address to binary:

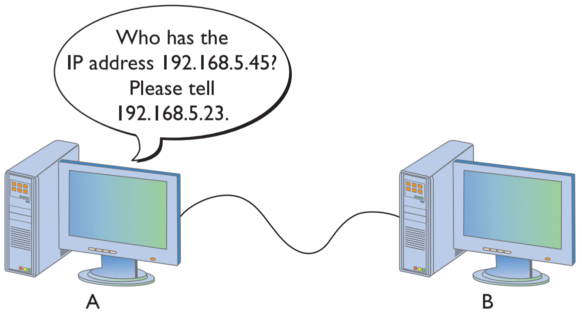
11000000101010000000010100101100

Computer A compares its IP address to Computer B's IP address using the subnet mask, as shown in Figure 7.16. For clarity, I've added a line to show you where the ones end and the zeroes begin in the subnet mask. Computers certainly don't need the pretty red line!


Computer A's IP: 11000000101010000000010100010111
Subnet mask 11111111111111111111111100000000
Computer B's IP: 11000000101010000000010100101100
These all match! It's a local call.

• Figure 7.16 Comparing addresses

Ah ha! Computer A's and Computer B's network IDs match! It's a local call. Knowing this, Computer A can now send out an ARP broadcast, as shown in Figure 7.17, to determine Computer B's MAC address. The **Address Resolution Protocol (ARP)** is how TCP/IP networks figure out the MAC addresses based on the destination IP address.

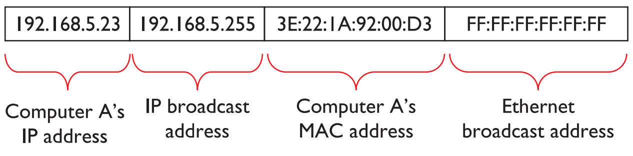

Who has the IP address 192.168.5.45? Please tell 192.168.5.23.
A
B

• Figure 7.17 Sending an ARP

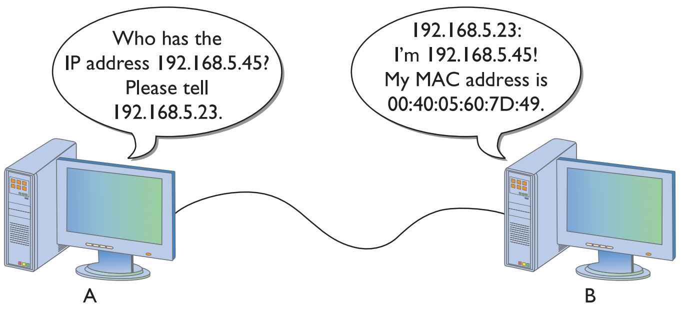
The addressing for the ARP packet looks like Figure 7.18. Note that Computer A's IP address and MAC address are included.

Computer B responds to the ARP by sending Computer A an ARP response (Figure 7.19). Once Computer A has Computer B's MAC address, it will now start sending packets.

But what happens when Computer A wants to send a packet to Computer C? First, Computer A compares Computer C's IP address to its own using the subnet mask (Figure 7.20). It sees that the IP addresses do not match in the ones part of the subnet mask—the network IDs don't match—meaning this is a long-distance call.



• Figure 7.18 ARP packet header showing addresses

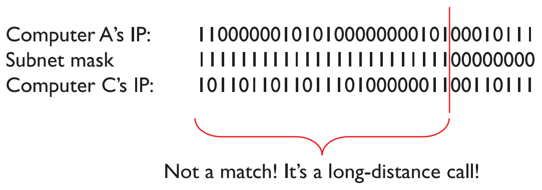

Who has the IP address 192.168.5.45? Please tell 192.168.5.23.
192.168.5.23: I'm 192.168.5.45! My MAC address is 00:40:05:60:7D:49.
A
B

• Figure 7.19 Computer B responds

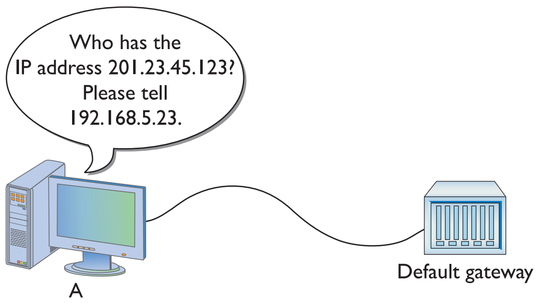
Cross Check

ARP and the OSI Model

Remember the OSI model from way back in Chapter 2? Which layer does IP work on? How about MAC addresses? Where do you think ARP fits into the OSI model?


Computer A's IP: 110000001010100000000101|00010111
Subnet mask 111111111111111111111111|00000000
Computer C's IP: 101101101101110100000011|00110111
Not a match! It's a long-distance call!

• **Figure 7.20** Comparing addresses again


Who has the IP address 201.23.45.123? Please tell 192.168.5.23.
A
Default gateway

• **Figure 7.21** Sending an ARP to the gateway

Whenever a computer wants to send to an IP address on another LAN, it knows to send the packet to the default gateway. It still sends out an ARP, but this time to the default gateway (Figure 7.21). Once Computer A gets the default gateway's MAC address, it then begins to send packets.

Subnet masks are represented in dotted decimal just like IP addresses—just remember that both are really 32-bit binary numbers. All of the following (shown in both binary and dotted decimal formats) can be subnet masks:

11111111111111111111111100000000 = 255.255.255.0
11111111111111110000000000000000 = 255.255.0.0
11111111000000000000000000000000 = 255.0.0.0

Most network folks represent subnet masks using special shorthand: a / character followed by a number equal to the number of ones in the subnet mask. Here are a few examples:

11111111111111111111111100000000 = /24 (24 ones)
11111111111111110000000000000000 = /16 (16 ones)
11111111000000000000000000000000 = /8 (8 ones)

Try This!

ARP in Windows

To show Windows' current ARP table, open a command line and type this:

```
arp –a
```

You should see results similar to this:

```
Interface: 192.168.4.71 --- 0x4
  Internet Address      Physical Address      Type
  192.168.4.76          00-1d-e0-78-9c-d5     dynamic
  192.168.4.81          00-1b-77-3f-85-b4     dynamic
```

Now delete one of the entries in the ARP table with the command:

```
arp –d [ip address from the previous results]
```

Run the `arp –a` command again. The line for the address you specified should be gone. Now try and PING the address you deleted and check the ARP table again. Is the deleted address back?

An IP address followed by the / and number tells you the IP address and the subnet mask in one statement. For example, 201.23.45.123/24 is an IP address of 201.23.45.123, with a subnet mask of 255.255.255.0. Similarly, 184.222.4.36/16 is an IP address of 184.222.4.36, with a subnet mask of 255.255.0.0.

Fortunately, computers do all of this subnet filtering automatically. Network administrators need only to enter the correct IP address and subnet mask when they first set up their systems, and the rest happens without any human intervention.

If you want a computer to work in a routed network (like the Internet), you absolutely must have an IP address that's part of its network ID, a subnet mask, and a default gateway. No exceptions!

By definition, all computers on the same network will have the same subnet mask and network ID.

Class IDs

The Internet is by far the biggest and the most complex TCP/IP network, numbering over half a billion computers at the beginning of 2009 and growing quickly. The single biggest challenge for the Internet is to make sure no two devices on the Internet share the same IP address. To support the dispersion of IP addresses, an organization called the **Internet Assigned Numbers Authority (IANA)** was formed to track and disperse IP addresses to those who needed them. Initially handled by a single person (the famous Jon Postel) until 1998, the IANA has grown dramatically and now oversees a number of Regional Internet Registries (RIRs) who parcel out IP addresses to large ISPs. The vast majority of end users get their IP addresses from their respective ISPs. The IANA passes out IP addresses in contiguous chunks called **class licenses**, outlined in the following table:

	First Decimal Value	Addresses	Hosts per Network ID
Class A	1–126	1.0.0.0–126.255.255.255	16,277,214
Class B	128–191	128.0.0.0–191.255.255.255	65,534
Class C	192–223	192.0.0.0–223.255.255.255	254
Class D	224–239	224.0.0.0–239.255.255.255	Multicast
Class E	240–255	240.0.0.0–255.255.255.255	Reserved

A typical Class A license, for example, would have a network ID that starts between 1–126; hosts on that network would have only the first octet in common, with any numbers for the other three octets. Having three octets to use for hosts means you can have an enormous number of possible hosts, over 16 million different number combinations. The subnet mask for Class A licenses is 255.0.0.0.

A Class B license, with a subnet mask of 255.255.0.0, uses the first two octets to define the network ID. This leaves two octets to define host IDs, which means each Class B network ID can have up to 65,534 different hosts.

A Class C license uses the first three octets to define only the network ID. All hosts in network 192.168.35.0, for example, would have all three first numbers in common. Only the last octet defines the host IDs, which leaves only 254 possible unique addresses. The subnet mask for Class C licenses is 255.255.255.0.

Make sure you memorize the IP class licenses! You should be able to look at any IP address and tell its class license. A trick to help: The first binary octet of a Class A address always begins with a 0 (0*xxxxxxx*); for Class B, it's 10 (10*xxxxxx*); for Class C, 110 (110*xxxxx*); Class D is 1110 (1110*xxxx*); and for Class E, it's 1111 (1111*xxxx*).

Multicast and reserved class licenses—Classes D and E, respectively—are rather strange and deserve some discussion. There are three types of ways to send a packet: a broadcast, which is where every computer on the LAN hears the message, a **unicast**, where one computer sends a message directly to another user, and **multicast**, where a single computer sends a packet to a group of interested computers. Multicast is uncommon on individual computers, but is often used when routers talk to each other. Reserved addresses are just that—reserved and never used except for occasional experimental reasons.

IP class licenses worked well for the first few years of the Internet, but quickly ran into trouble due to the fact that they didn't quite fit for everyone. Early on IANA gave away IP class licenses rather generously, perhaps too generously. Over time, unallocated IP addresses became scarce. Additionally, the IP class licenses concept didn't scale well. If an organization needed 2000 IP addresses, for example, it either had to take a single Class B license (wasting 63,000 addresses) or eight Class C licenses. As a result, a new method of generating blocks of IP addresses, called **Classless Inter-Domain Routing (CIDR)**, was developed.

CIDR and Subnetting

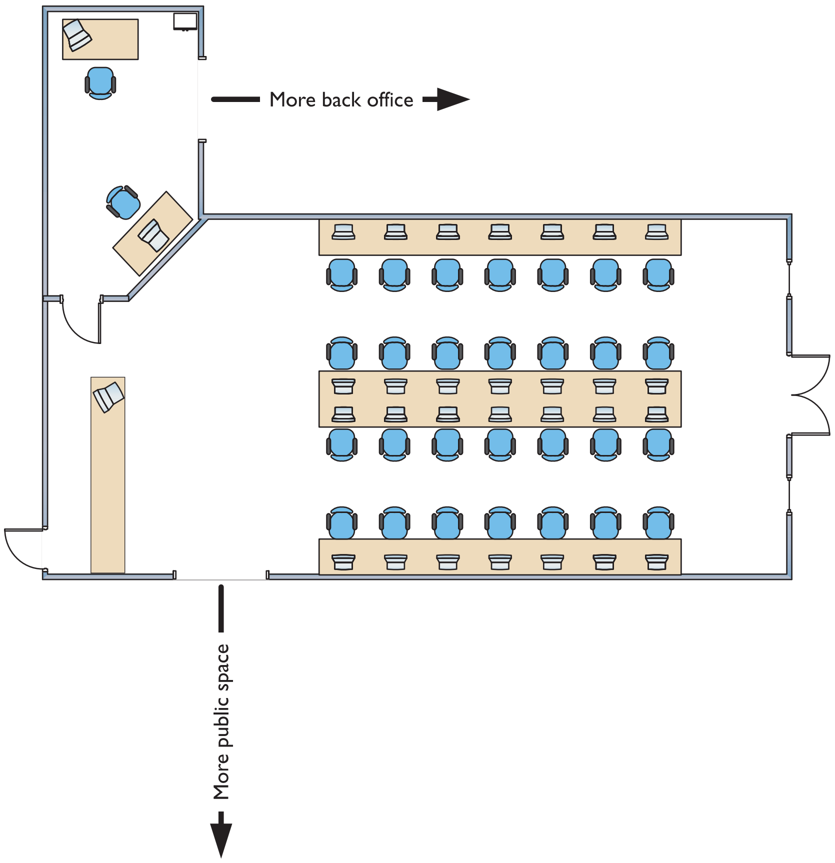
CIDR is based on a concept called **subnetting**: taking a single class of IP addresses and chopping it up into multiple smaller groups. CIDR and subnetting are virtually the same thing. Subnetting is done by an organization—it is given a block of addresses and then breaks the single block of addresses into multiple subnetworks. CIDR is done by an ISP—it is given a block of addresses, subnets the block into multiple subnets, and then passes out the smaller individual subnets to customers. Subnetting and CIDR have been around for quite a long time now and are a critical part of all but the smallest TCP/IP networks. Let's first discuss subnetting and then visit CIDR.

Subnetting

You need to know how to subnet to pass the CompTIA Network+ exam.

Subnetting enables a much more efficient use of IP addresses than does using class licenses. It also enables you to separate a network for security (separating a bank of public access computers from your more private computers) and for bandwidth control (separating a heavily used LAN from one that's not so heavily used).

The cornerstone to subnetting lies in the subnet mask. You take an existing /8, /16, or /24 subnet and extend the subnet mask by adding more ones (and taking away the same number of zeroes). For example, let's say you have an Internet café with about 50 computers, 40 of which are for public use and 10 of which are used in the back office for accounting and such (Figure 7.22). Your network ID is 192.168.4/24. You want to prevent people using the public systems from accessing your private machines, so you decide to do a subnet. You also have wireless Internet and want to separate wireless clients (never more than 10) on their own subnet.



• **Figure 7.22** Layout of the network

There are two items to note about subnetting. First, start with the given subnet mask and move it to the right until you have the number of subnets you need. Second, forget the dots. Never try to subnet without first converting to binary. Too many techs are what I call "victims of the dots." They are so used to working only with class licenses that they forget there's more to subnets than just /8, /16, and /24 networks. There is no reason network IDs must end on the dots. The computers, at least, think it's perfectly fine to have subnets that end at points between the periods, such as /26, /27, or even /22. The trick here is to stop thinking about network IDs, and subnet masks just in their dotted decimal format, and instead go back to thinking of them as binary numbers.

Let's begin subnetting the café's network of 192.168.4/24. Start by changing a zero to a one on the subnet mask so the /24 becomes a /25 subnet:

```
11111111111111111111111110000000
```

Calculating Hosts

Before we even go one step further you need to answer this question: On a /24 network, how many hosts can you have? Well, if you used dotted decimal notation you might say

192.168.4.1 to 192.168.4.254 = 254 hosts

But do this from the binary instead. In a /24 network you have eight zeroes that can be the host ID:

00000001 to 11111110 = 254

There's a simple piece of math here: $2^{(\text{number of zeroes in the subnet mask})} - 2$

$2^8 - 2 = 254$

If you remember this simple formula, you can always determine the number of hosts for a given subnet. This is critical! Memorize this!

If you have a /16 subnet mask on your network, what is the maximum number of hosts you can have on that network?

1. Since a subnet mask always has 32 digits, a /16 subnet means you have 16 zeroes left after the 16 ones.
2. $2^{16} - 2$ = 65,534 total hosts.

If you have a /26 subnet mask on your network, what is the maximum number of hosts you can have on that network?

1. Since a subnet mask always has 32 digits, a /26 subnet means you have 6 zeroes left after the 26 ones.
2. $2^6 - 2$ = 62 total hosts.

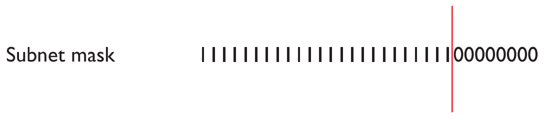
Excellent! Knowing how to determine the number of hosts for a particular subnet mask will help you tremendously in a moment.

Your First Subnet

Subnetting only truly makes sense when you do it in binary!

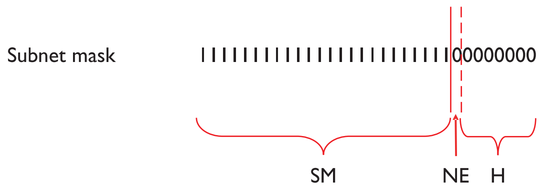
Let's now make a subnet. All subnetting begins with a single network ID. In this scenario, you need to convert the 192.168.4/24 network ID for the café into three network IDs: one for the public computers, one for the private computers, and one for the wireless clients.

The primary tool to subnet is the existing subnet mask. Write it out in binary. Place a line at the end of the ones as shown in Figure 7.23.



• Figure 7.23 Step 1 in subnetting

Now draw a second line one digit to the right, as shown in Figure 7.24. You've now separated the subnet mask into three areas that I call (from left to right) the subnet mask (SM), the network ID extension (NE), and the hosts (H). These are not industry terms so you won't see them on the CompTIA Network+ exam, but they're a handy Mike Trick that makes the process of subnetting a lot easier.



• Figure 7.24 Organizing the subnet mask

You now have a /25 subnet mask. At this point, most people first learning how to subnet start to freak out. They're challenged by the idea that a subnet mask of /25 isn't going to fit into one of the three pretty subnets of 255.0.0.0, 255.255.0.0, or 255.255.255.0. They think, "That can't be right! Subnet masks are made out of only 255s and 0s." That's not correct. A subnet mask is a string of ones followed by a string of zeroes. People only convert it into dotted decimal to enter things into computers. So convert /25 into dotted decimal. First write out 25 ones, followed by seven zeroes. (Remember, subnet masks are *always* 32 binary digits long.)

```
11111111111111111111111110000000
```

Put the periods in between every eight digits:

```
11111111.11111111.11111111.10000000
```

Then convert them to dotted decimal:

```
255.255.255.128
```

Get used to the idea of subnet masks that use more than 255s and 0s. Here are some examples of perfectly legitimate subnet masks. Try converting these to binary to see for yourself.

```
255.255.255.224
255.255.128.0
255.248.0.0
```

Calculating Subnets

When you subnet a network ID, you need to follow the rules and conventions dictated by the good folks who developed TCP/IP to ensure that your new subnets can interact properly with each other and with larger networks. The rules to subnetting are as follows:

1. Starting with a beginning subnet mask, you extend the subnet extension until you have the number of subnets you need.
2. You cannot have an NE of all zeroes or all ones, so you calculate the number of subnets using this formula: new subnets = $2^{(\text{number of network ID extension digits})} - 2$.

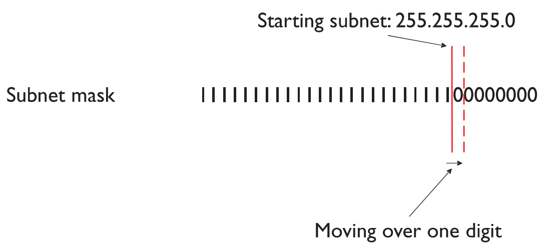
Rule number two can trip up folks new to subnetting, because it seems arbitrary, but it's not. What defines the subnets of a network ID are the different combinations of binary numbers within the NE. All zeroes is by definition a network ID; all ones is used only for broadcasting, so that's not allowed either.

Let's practice this a few times. Figure 7.25 shows a starting subnet of 255.255.255.0. If we move the network ID extension over one, it's only a single digit.

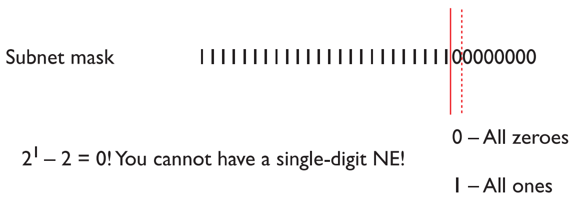
That single digit is only a zero or a one, and you can't have only a zero or a one for a network ID extension! So let's add a third rule:

3. You cannot have a single-character network ID extension. You always start by moving the subnet at least two digits (Figure 7.26).

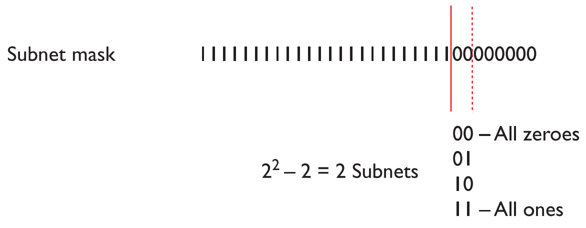
Back to the first subnet. Let's take /24 and subnet it down to /26. Extending the network ID by two digits creates four new network IDs (two of which you won't be able to use). To see each of these network IDs, first convert the original network ID—192.168.4.0—into binary. Then add the four different network ID extensions to the end, as shown in Figure 7.27.



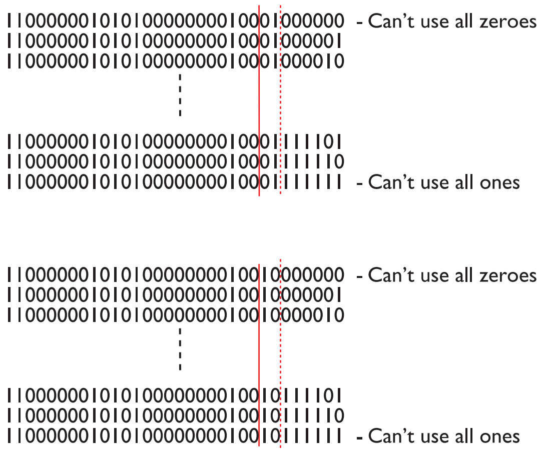
● Figure 7.25 Organizing the subnet mask



● Figure 7.26 Single-digit network ID extensions are not allowed.



• Figure 7.27 Creating the new network IDs



• Figure 7.28 New network ID address ranges

Since the network ID extension can't be all zeroes and all ones, you only get two new network IDs. Figure 7.28 shows all of the IP addresses for each of the two new network IDs.

Now convert these two network IDs back to dotted decimal:

192.168.4.64/26 (192.168.4.65 – 192.168.4.126)
192.168.4.128/26 (192.168.4.129 – 192.168.4.191)

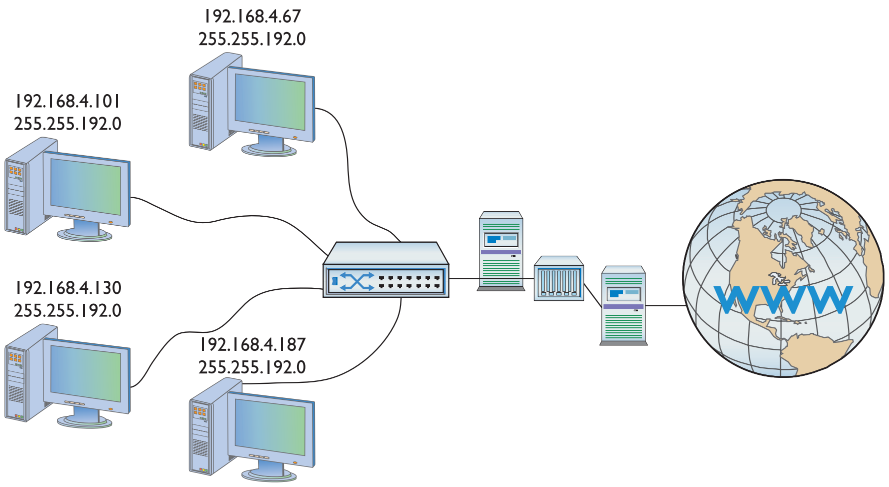
Congratulations! You've just taken a single network ID, 192.168.4.0/24, and subnetted it into two new network IDs! Figure 7.29 shows how you can use these two network IDs in a network.

There's only one problem—the café needs three subnets, not just two! So let's first figure out how large of a network ID extension is needed:

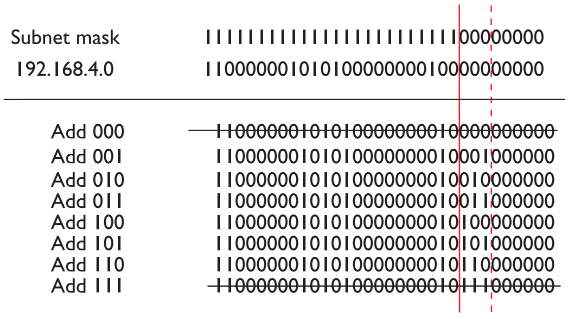
- Two NE digits = $2^2 - 2 = 2$ network IDs
- Three NE digits = $2^3 - 2 = 6$ network IDs

Okay, you need to extend the NE three digits to get six network IDs. Because the café only needs three, three are wasted—welcome to subnetting.

If wasting subnets seems contrary to the goal of efficient use, keep in mind that subnetting has two goals: efficiency and making multiple network IDs from a single network ID. This example is geared more toward the latter goal.



• Figure 7.29 Two networks using the two network IDs


Subnet mask 11111111111111111111111100000000
192.168.4.0 11000000101010000000010000000000
Add 000 ~~11000000101010000000010000000000~~
Add 001 11000000101010000000010000100000
Add 010 11000000101010000000010001000000
Add 011 11000000101010000000010001100000
Add 100 11000000101010000000010010000000
Add 101 11000000101010000000010010100000
Add 110 11000000101010000000010011000000
Add 111 ~~11000000101010000000010011100000~~

● **Figure 7.30** Moving the network ID extension three digits

First, move the NE over three digits. This creates a /27 subnet for all the new network IDs (Figure 7.30).

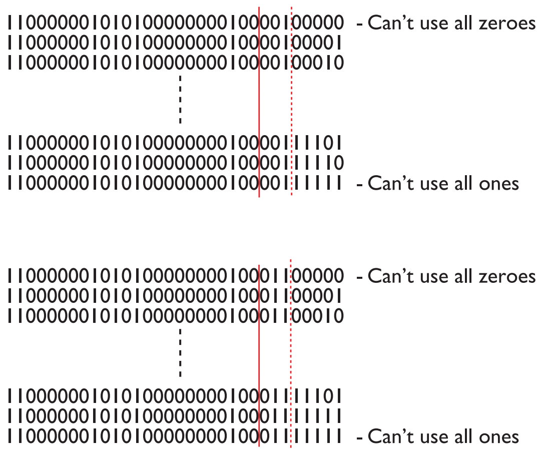
To help you visualize the address range, I'll calculate the first two subnets—using 001 and 011 (Figure 7.31). Please do the other four for practice.

Note that in this case you only get $2^5 - 2 = 30$ hosts per network ID! These better be small networks!

Converting these to dotted decimal we get:

192.168.4.32/27 (192.168.4.33 – 192.168.4.62)
192.168.4.64/27 (192.168.4.65 – 192.168.4.94)
192.168.4.96/27 (192.168.4.97 – 192.168.4.126)
192.168.4.128/27 (192.168.4.129 – 192.168.4.158)
192.168.4.160/27 (192.168.4.161 – 192.168.4.190)
192.168.4.192/27 (192.168.4.193 – 192.168.4.222)

These two examples started with a Class C address. There's no reason not to start with any starting network ID. Nothing changes from the process you just learned.



● **Figure 7.31** Two of the six network ID address ranges

Tech Tip

ISPs and Classless Addresses

If you order real, unique, ready-for-the-Internet IP addresses from your local ISP, you'll invariably get a classless set of IP addresses. More importantly, when you order them for clients, you need to be able to explain why their subnet mask is 255.255.255.192, when all the books they read tell them it should be 255.255.255.0!

Manual Dotted Decimal to Binary Conversion

The best way to convert from dotted decimal to binary and back is using a calculator. It's easy, fast, and accurate. There's always a chance, however, that you may find yourself in a situation where you need to convert without a calculator. Fortunately, manual conversion, while a bit tedious, is also fairly easy. The secret is to simply remember a single number: 128.

Take a piece of paper and write the number 128 in the top-left corner. Now, what is half of 128? That's right, 64. Write 64 next to 128. Now keep dividing the previous number in half until you get to the number one. The result will look like this:

```
128   64    32    16    8    4    2    1
```

Notice that you have eight numbers. Each of these numbers corresponds to a position of one of the eight binary digits. To convert an 8-bit value to dotted decimal, just take the binary value and put the numbers under the corresponding eight digits. Wherever there's a one, add that decimal value.

Let's take the binary value 10010110 into decimal. Write down the numbers as just shown, then write the binary values underneath each corresponding decimal number:

```
128    64     32     16     8     4     2     1
  1     0      0      1     0     1     1     0
```

Add the decimal values that have a 1 underneath:

128+16+4+2 = 150

Converting from decimal to binary is a bit more of a challenge. You still start with a line of decimal numbers starting with 128, but this time place the decimal value above. If the number you're trying to convert is greater than or equal to the number underneath, subtract it and place a 1 underneath that value. If not, then place a 0 underneath and move the number to the next position to the right. Let's give this a try by converting 221 to binary. Begin by placing 221 over the 128:

```
221
128    64    32    16     8     4     2     1
 93
  1
```

Now place the remainder, 93, over the 64:

```
       93
128    64    32    16     8     4     2     1
       29
  1     1
```

Place the remainder, 29, over the 32. The number 29 is less than 32, so place 0 underneath the 32 and move to 16:

```
                   29
128    64    32    16     8     4     2     1
                   13
  1     1     0     1
```

Then move to the 8:

```
                         13
128    64    32    16     8     4     2     1
                          5
  1     1     0     1     1
```

Then the 4:

```
                                5
128    64    32    16     8     4     2     1
                                1
  1     1     0     1     1     1
```

Then the 2. The number 1 is less than 2, so drop a 0 underneath and move to 1:

```
                                            1
128    64    32    16     8     4     2     1

  1     1     0     1     1     1     0     1
```

Finally, the 1; 1 is equal to 1, so put a 1 underneath and you're done. The number 221 in decimal is equal to 11011101 in binary.

Make sure you can manually convert decimal to binary, and binary to decimal.

CIDR: Subnetting in the Real World

I need to let you in on a secret—there's a better than average chance that you'll never have to do subnetting in the real world. That's not to say that subnetting isn't important. It's a critical part of the structure of the Internet. There are two situations in which subnetting most commonly takes place: ISPs who receive class licenses from IANA and then subnet those class licenses for customers, and very large customers who take subnets (sometimes already subnetted class licenses from ISPs) and make their own subnets. Even if you'll never make a working subnet in the real world, there are a number of reasons to learn subnetting.

First and most obvious, the CompTIA Network+ exam expects you to know subnetting. You need to take any existing network ID and break it down into a given number of subnets. You need to know how many hosts the resulting network IDs possess. You need to be able to calculate the IP addresses and the new subnet masks for each of the new network IDs.

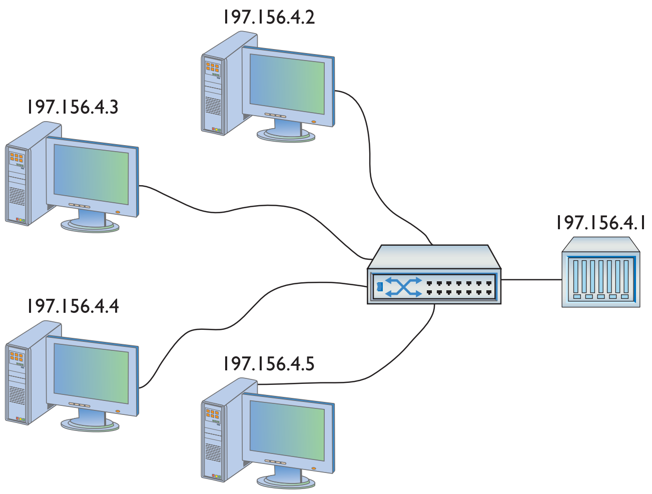
Second, even if you never do your own subnetting, there's a pretty good chance that you'll contact an ISP and get CIDR addresses. You can't think about subnet masks in terms of dotted decimal. You need to think of subnets in terms of CIDR values like /8, /22, /26, and so on.

Third, there's a better than average chance you'll look to more advanced IT certifications. Most Cisco, many Microsoft, and a large number of other certifications assume you understand subnetting. It's a competency standard that everyone who's serious about networking understands in detail, a clear separation of those who know networks from those who do not.

You've done well, my little padawan. Subnetting takes a little getting used to. Go take a break. Take a walk outside. Play some World of Warcraft. (You can find me on Blackwater Raiders. My current favorite character is "Polope," undead.) Or fire up your Steam client and see if I'm playing Counter-Strike or Left4Dead (player name "desweds"). After a good mental break, dive back into subnetting and *practice*. Take any old network ID and practice making multiple subnets—lots of subnets!

Using IP Addresses

Whew! After all that subnetting, you've reached the point where it's time to start actually using some IP addresses. That is, after all, the goal of going through all that pain. There are two ways to give a computer an IP address, subnet mask, and default gateway: either by typing in all the information (called **static addressing**) or by having some server program running on a system that automatically passes out all the IP information to systems as they boot up on or connect to a network (called **dynamic addressing**). Additionally, you must learn about a number of specialty IP addresses that have unique meanings in the IP world to make this all work.



• **Figure 7.32** A small network

Static IP Addressing

Static addressing means typing all of the IP information into each of your clients. But before you type in anything, you have to answer two questions: what are you typing in and where do you type it? Let's visualize a four-node network like the one shown in Figure 7.32.

To make this network function, each computer must have an IP address, a subnet mask, and a default gateway. First, decide what network ID to use. In the old days, you were given a block of IP addresses from your ISP to use. Assume that's still the method and you've been allocated a Class C license for 197.156.4/24.

The first rule of Internet addressing is . . . no one talks about Internet addressing. Actually we can maul the *Fight Club* reference and instead say, "The first rule of Internet addressing is that you can do whatever you want with your own network ID." There are no rules other than to make sure every computer gets a legit IP address and subnet mask for your network ID and make sure every IP address is unique. You don't have to use the numbers in order, you don't have to give the default gateway the 192.156.4.1 address—you can do it any way you want. That said, most networks follow a common set of principles:

1. Give the default gateway the first IP address in the network ID.
2. Try to use the IP addresses in some kind of sequential order.
3. Try to separate servers from clients. For example, servers could have the IP addresses 197.156.4.10 to 197.156.4.19, while the clients range from 197.156.4.200 to 197.156.4.254.
4. Write down whatever you choose to do so the person who comes after you understands.

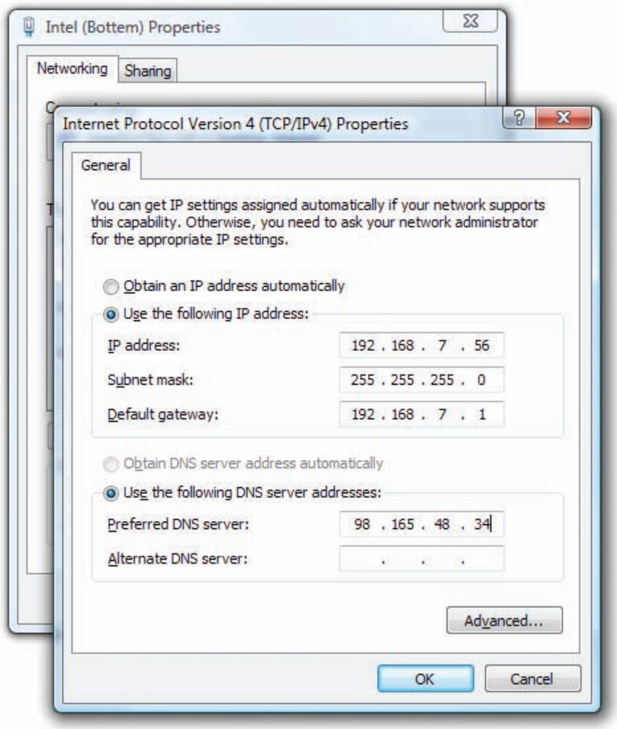
These principles have become unofficial standards for network techs, and following them will make you very popular with whoever has to manage your network in the future.

Now you can give each of the computers an IP address, subnet mask, and default gateway.

Every operating system has some method for you to enter in the static IP information. In Windows, you use the Internet Protocol Version 4 (TCP/IPv4) Properties dialog box, as shown in Figure 7.33.

On a Macintosh OS X, run the Network utility in System Preferences to enter in the IP information (Figure 7.34).

The only universal tool for entering IP information on UNIX/Linux systems is the command-line IFCONFIG command, as shown in Figure 7.35. A warning about setting static IP addresses with IFCONFIG: any address entered will not be permanent and will be lost on reboot. To make the new IP permanent you need to find and edit your network configuration files. Fortunately, modern distributions (distros) make your life a bit easier. Almost every flavor of UNIX/Linux comes with some handy graphical program, such as Network Configuration in the popular Ubuntu Linux distro (Intrepid Ibex 8.10) (Figure 7.36).


Intel (Bottom) Properties
Networking
Sharing
Internet Protocol Version 4 (TCP/IPv4) Properties
General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

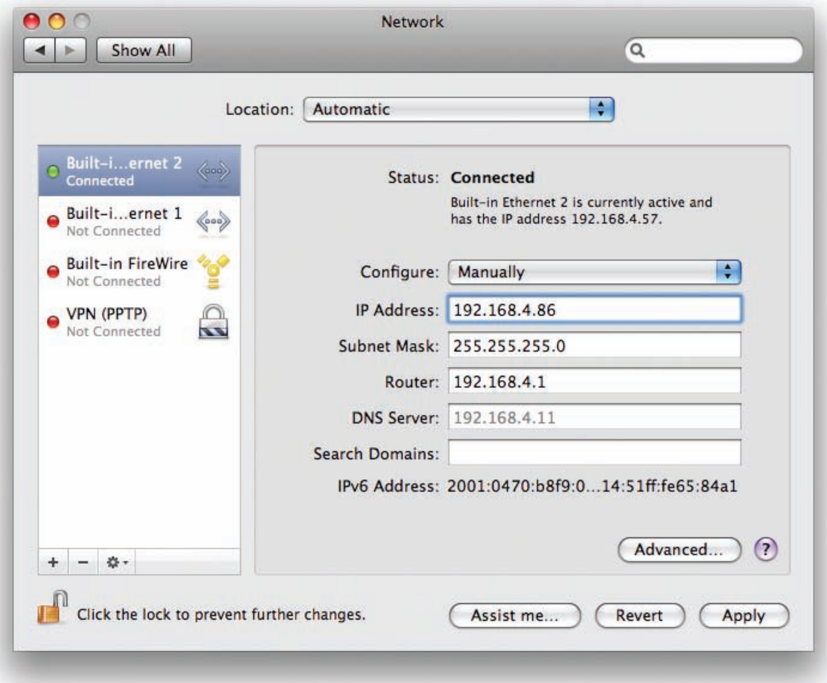
Obtain an IP address automatically [✓ Use the following IP address:]

IP address:	192 . 168 . 7 . 56
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	192 . 168 . 7 . 1

Obtain DNS server address automatically [✓ Use the following DNS server addresses:]

Preferred DNS server:	98 . 165 . 48 . 34
Alternate DNS server:	. . .

• **Figure 7.33** Entering static IP information in Windows Internet Protocol Version 4 (TCP/IPv4) Properties



Location: Automatic

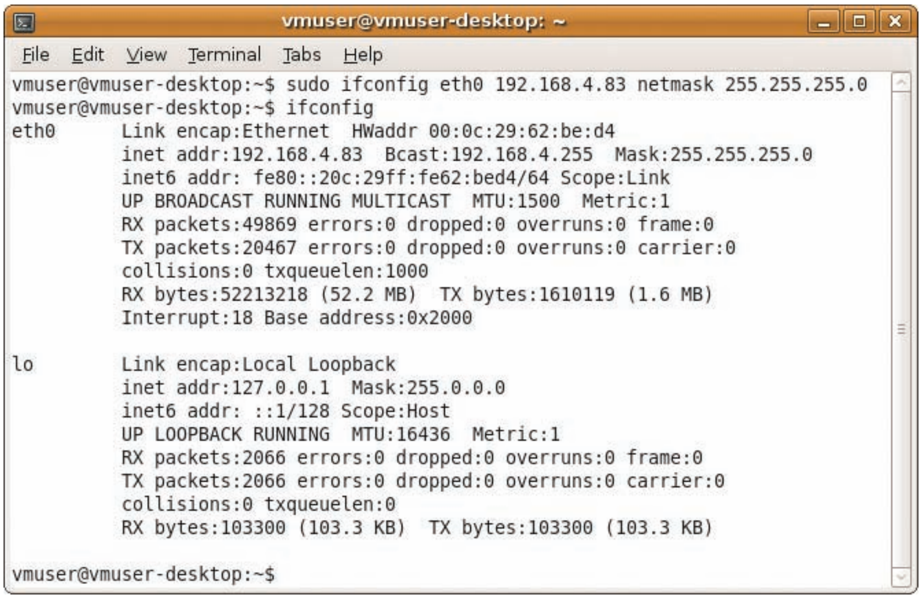
Status: **Connected**

Built-in Ethernet 2 is currently active and has the IP address 192.168.4.57.

Configure:	Manually
IP Address:	192.168.4.86
Subnet Mask:	255.255.255.0
Router:	192.168.4.1
DNS Server:	192.168.4.11
Search Domains:	
IPv6 Address:	2001:0470:b8f9:0...14:51ff:fe65:84a1

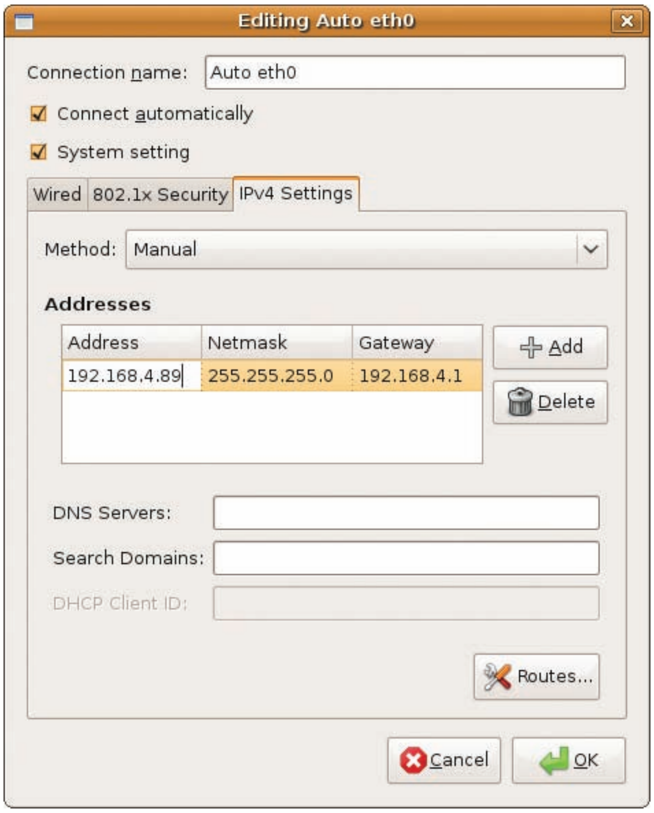
Click the lock to prevent further changes.

• **Figure 7.34** Entering static IP information in the OS X Network utility



```
vmuser@vmuser-desktop:~$ sudo ifconfig eth0 192.168.4.83 netmask 255.255.255.0
vmuser@vmuser-desktop:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:62:be:d4
          inet addr:192.168.4.83  Bcast:192.168.4.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe62:bed4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:49869 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20467 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:52213218 (52.2 MB)  TX bytes:1610119 (1.6 MB)
          Interrupt:18 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2066 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2066 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:103300 (103.3 KB)  TX bytes:103300 (103.3 KB)

vmuser@vmuser-desktop:~$
```

• **Figure 7.35** IFCONFIG command to set static IP address



Connection name: Auto eth0

☑ Connect automatically

☑ System setting

Method: Manual

Addresses

Address	Netmask	Gateway
192.168.4.89	255.255.255.0	192.168.4.1

DNS Servers:

Search Domains:

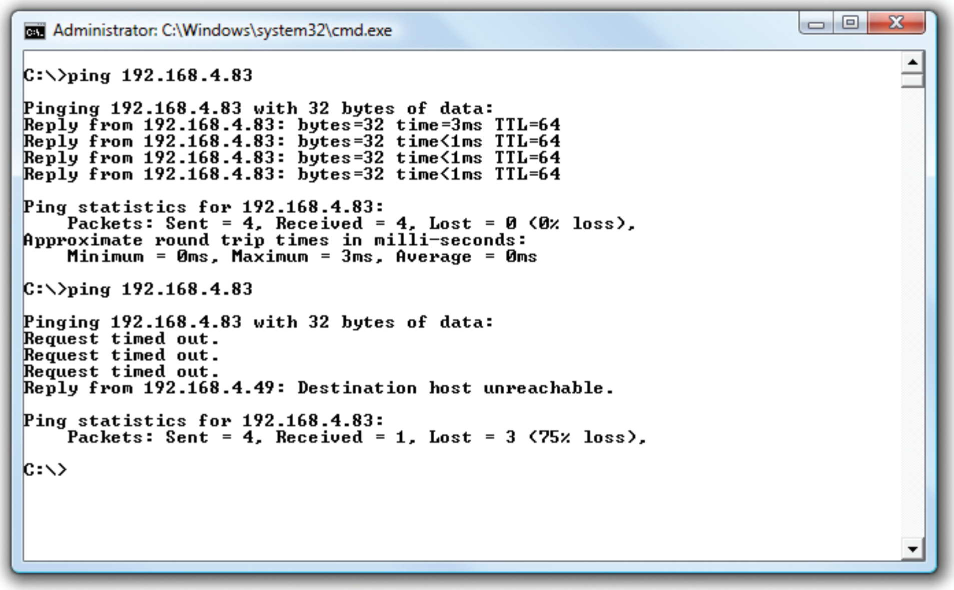
DHCP Client ID:

• **Figure 7.36** Ubuntu's Network Configuration utility

Once you've added the IP information to at least two systems, you should always verify by using the PING command as shown in Figure 7.37. Always verify with PING—it's too easy to make a typo when you use static IP addresses.

If you set an IP address and your PING is not successful, first check your IP settings. Odds are good you made a typo. Otherwise, check your connections, driver, and so forth.

Static addressing has been around for a long time and is still heavily used for more critical systems on your network, but static addressing poses one big problem: it's a serious pain to make any changes to the network. Most systems today use a far easier and flexible method to get their IP information: dynamic IP addressing.



```
C:\>ping 192.168.4.83

Pinging 192.168.4.83 with 32 bytes of data:
Reply from 192.168.4.83: bytes=32 time=3ms TTL=64
Reply from 192.168.4.83: bytes=32 time<1ms TTL=64
Reply from 192.168.4.83: bytes=32 time<1ms TTL=64
Reply from 192.168.4.83: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.4.83:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 3ms, Average = 0ms

C:\>ping 192.168.4.83

Pinging 192.168.4.83 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Reply from 192.168.4.49: Destination host unreachable.

Ping statistics for 192.168.4.83:
    Packets: Sent = 4, Received = 1, Lost = 3 (75% loss),

C:\>
```

• **Figure 7.37** Two PINGs (successful PING on top, unsuccessful PING on bottom)

BOOTP is common on UNIX/Linux and Macintosh OS X systems.

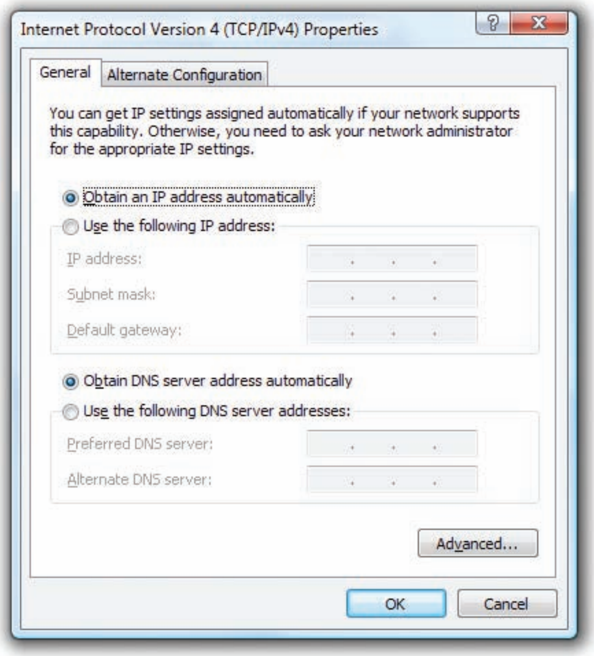
Dynamic IP Addressing

Dynamic IP addressing, better known as **Dynamic Host Configuration Protocol (DHCP)** or the older (and much less popular) **Bootstrap Protocol (BOOTP)**, automatically assigns an IP address whenever a computer connects to the network. DHCP (and BOOTP, but for simplicity I'll just say DHCP) works in a very simple process. First, a computer is configured to use DHCP. Every OS has some method to tell the computer to use DHCP, like the Windows example shown in Figure 7.38.

How DHCP Works

Once a computer is configured to use DHCP, it's called a DHCP client. When a DHCP client boots up it automatically sends out a special DHCP discovery packet using the broadcast address. This DHCP discovery message asks: "Are there any DHCP servers out there?" (See Figure 7.39.)

For DHCP to work, there must be one system on the LAN running special DHCP server software. This server is designed to respond to DHCP discovery requests with a DHCP offer. The DHCP server is configured to pass out IP addresses from a range (called a DHCP scope), a subnet

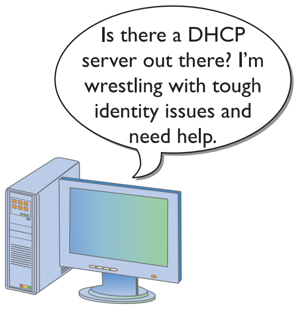


You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

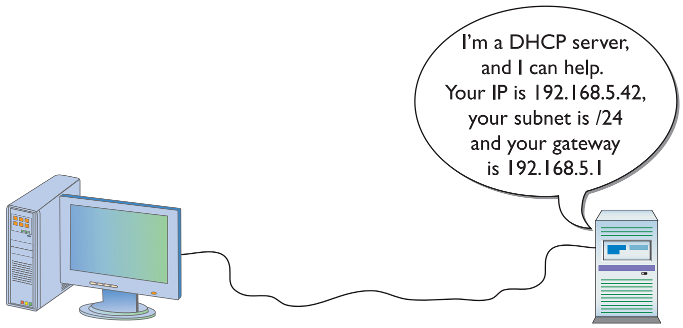
[✓ Obtain an IP address automatically]
Use the following IP address:

[✓ Obtain DNS server address automatically]
Use the following DNS server addresses:

• **Figure 7.38** Setting up for DHCP


Is there a DHCP server out there? I'm wrestling with tough identity issues and need help.

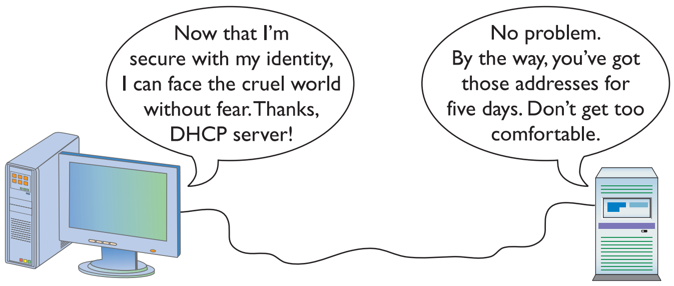
• Figure 7.39 Computer sending out a DHCP discovery message


I'm a DHCP server, and I can help. Your IP is 192.168.5.42, your subnet is /24 and your gateway is 192.168.5.1

• Figure 7.40 DHCP server sending DHCP offer

mask, and a default gateway (Figure 7.40). It also passes out other information that we'll cover in later chapters.

At this point the DHCP client sends out a DHCP request—a poor name choice as it really is accepting the offer. The DHCP server then sends a DHCP acknowledge and lists the MAC address as well as the IP information given to the DHCP client in a database (Figure 7.41).


Now that I'm secure with my identity, I can face the cruel world without fear. Thanks, DHCP server!
No problem. By the way, you've got those addresses for five days. Don't get too comfortable.

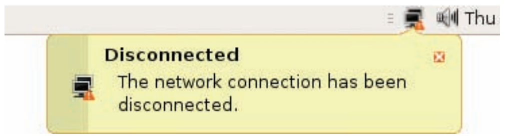
• Figure 7.41 DHCP request and DHCP acknowledge

The acceptance from the DHCP client of the DHCP server's data is called a **DHCP lease**. A DHCP lease is set for a fixed amount of time, usually 5 to 8 days. At the end of the lease time, the DHCP client simply makes another DHCP discovery message. The DHCP server looks at the MAC address information and, unless another computer has taken the lease, will always give the DHCP client the same IP information, including the same IP address.

Living with DHCP

DHCP is very convenient and, as such, very popular. So popular that it's very rare to see a user's computer on any network using static addressing.

It's important to know how to deal with the problems that arise with DHCP. The single biggest issue is when a DHCP client tries to get a DHCP address and fails. It's easy to tell when this happens because the operating system will post some form of error telling you there's a problem (Figure 7.42) and the DHCP client will have a rather strange address in the 169.254/16 network ID.



Disconnected
The network connection has been disconnected.

• Figure 7.42 DHCP error in Ubuntu Linux

Systems that use static IP addressing can never have DHCP problems.

This special IP address is generated by **Automatic Private IP Addressing (APIPA)**. All DHCP clients are designed to generate an APIPA address automatically if there's no response to a DHCP discovery message. The client generates the last two octets of an APIPA address automatically. This will at least allow all the DHCP clients on a single network to continue to communicate with each other because they are on the same network ID. Unfortunately, there's no way for APIPA to give a default gateway, so you'll never get on the Internet using APIPA. That provides a huge clue to a DHCP problem: you can communicate with other computers on your network, but you can't get out to the Internet.

If you can't get out to the Internet, use whatever tool your OS provides to check your IP address. If it's an APIPA address, you instantly know you have a DHCP problem. First of all, try to reestablish the lease manually. Every OS has some way to do this. In Windows, you can type the following command:

```
ipconfig /renew
```

On a Macintosh, you can go to System Preferences and use the Network utility (Figure 7.43).

Sometimes you might find yourself in a situation where your computer gets confused and won't grab an IP address no matter what you try. In these cases you should first force the computer to release its lease. In Windows, get to a command prompt and type these two commands, each followed by pressing ENTER:

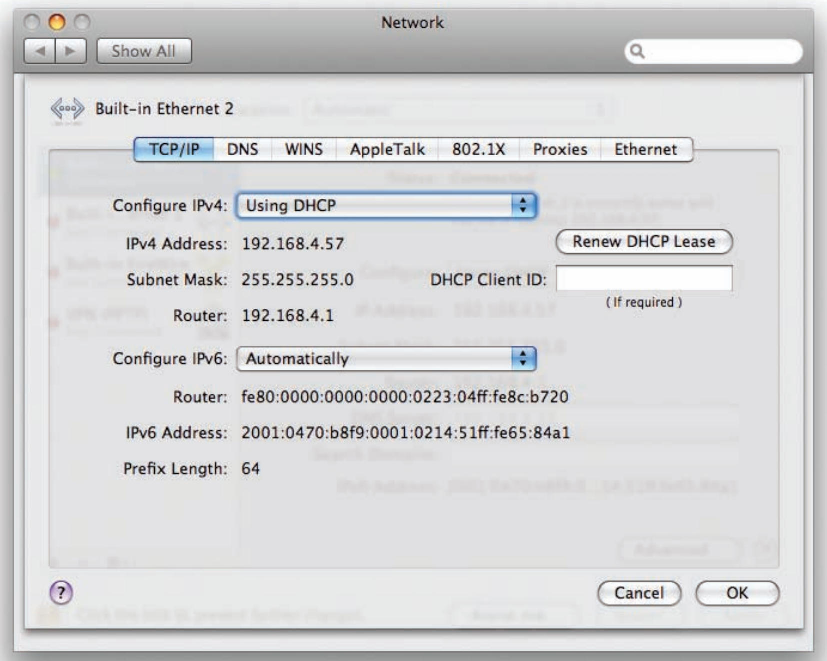
```
ipconfig /release
ipconfig /renew
```

In UNIX/Linux and even Macintosh you can use the IFCONFIG command to release and renew your DHCP address. Here's the syntax to release:

```
sudo ifconfig eth0 down
```

And here is the syntax to renew:

```
sudo ifconfig eth0 up
```



Configure IPv4: Using DHCP
IPv4 Address: 192.168.4.57
Subnet Mask: 255.255.255.0
Router: 192.168.4.1
DHCP Client ID:
Configure IPv6: Automatically
Router: fe80:0000:0000:0000:0223:04ff:fe8c:b720
IPv6 Address: 2001:0470:b8f9:0001:0214:51ff:fe65:84a1
Prefix Length: 64

• Figure 7.43 Network utility in System Preferences

Tech Tip

Case Matters

With UNIX, Linux, and Macintosh OS X command-line commands, case matters. If you run `sudo ifconfig eth0 down` *all in lowercase, for example, your Ethernet connection will drop as the DHCP or BOOTP lease is released. If you try running the same command all in upper case, on the other hand, the Linux et al. command prompt will look at you quizzically and then snort with derision. "What's this SUDO of which you speak?" it would say. And then give you a prompt for a "real" command. Watch your case with UNIX/Linux/OS X!*

Make sure you know how to configure your computers to use static IP addressing and know that you use PING to ensure they communicate. For dynamic IP addressing, make sure you know it has two names: DHCP and BOOTP. Understand that each client must have some way to "turn on" DHCP. Also understand the concept of a DHCP client and a DHCP server. Last, be comfortable with APIPA and releasing and renewing a lease on a client.

Even though by convention we use 127.0.0.1 as the loopback address, the entire 127.0.0.0/8 subnet is reserved for loopback! You can use any address in the 127.0.0.0/8 subnet as a loopback address.

Make sure you can quickly tell the difference between a private and a public IP address for the CompTIA Network+ exam.

Depending on your distribution, you may not need to type `sudo` first, but you will need to have root privileges to use IFCONFIG. Root privileges are Linux's version of administrative privileges in Windows.

Special IP Addresses

The folks who invented TCP/IP created a number of special IP addresses you need to know about. The first special address is 127.0.0.1, the **loopback address**. When you tell a device to send data to 127.0.0.1, you're telling that device to send the packets to itself. The loopback address has a number of uses. One of the most common is to use it with the PING command. We use the command PING 127.0.0.1 to test a NIC's capability to send and receive packets.

Lots of folks use TCP/IP in networks that either aren't connected to the Internet or want to hide their computers from the rest of Internet. Certain groups of IP addresses, known as private IP addresses, are available to help in these situations. All routers destroy private IP addresses. Those addresses can never be used on the Internet, making them a handy way to hide systems. Anyone can use these private IP addresses, but they're useless for systems that need to access the Internet—unless you use the mysterious and powerful NAT, which we will discuss in the next chapter. (Bet you're dying to learn about NAT now!) For the moment, however, let's just look at the ranges of addresses that are designated private IP addresses:

- 10.0.0.0 through 10.255.255.255 (1 Class A license)
- 172.16.0.0 through 172.31.255.255 (16 Class B licenses)
- 192.168.0.0 through 192.168.255.255 (256 Class C licenses)

All other IP addresses are public IP addresses.

Chapter 7 Review

Chapter Summary

After reading this chapter and completing the exercises, you should understand the following about TCP/IP.

Describe how the Internet Protocol works

- While MAC addresses are physical addresses burned into hardware, IP addresses are logical and are assigned via software.
- An IP address consists of 32 binary digits, often written in dotted decimal notation to make it easier for humans to read.
- Every IP address must be unique on its network.
- The utilities IPCONFIG (Windows) and IFCONFIG (UNIX/Linux/Macintosh) can be used to view IP address information.
- Every IP address contains both a network ID and a host ID. Computers on the same network will have the same network ID portion of an IP address while the host ID portion will be unique.
- The IP address of a network's router is called the default gateway. The router uses an internal routing table and network IDs to determine where to send network packets.
- A subnet mask helps to define the network ID of an IP address. All computers on a specific network share the same subnet mask.
- An Address Resolution Protocol (ARP) broadcast is used to determine the MAC address of the destination computer based on its IP address.
- Subnet masks are often written with the IP address in slash notation, such as 201.23.45.123/24. In this example, the IP address is 201.23.45.123 and the subnet mask consists of 24 ones, or 11111111.11111111.11111111.00000000 (255.255.255.0).
- The Internet Assigned Numbers Authority (IANA) is the organization responsible for tracking and dispersing IP addresses to Internet service providers.
- A broadcast is sent to every computer on the network. A unicast is sent from one computer to one other computer. A multicast is sent from one computer to multiple other computers.

Explain CIDR and subnetting

- Subnet masks enable network adapters to determine whether incoming packets are being sent to a local network address or a remote network.
- A subnet mask is similar in form to an IP address. Subnet masks consist of some number of ones, followed by zeroes, to make a total of 32 bits.
- Subnetting is done by organizations when they need to create subnetworks.
- Classless Inter-Domain Routing (CIDR) is when an ISP subnets a block of addresses and passes them out to smaller customers.
- Computers use subnet masks to distinguish (sub)network IDs from host IDs. Any number on the full IP address that corresponds to a 1 on the subnet mask is part of the network ID. Any uncovered (turned off or = "0") bits show the host ID of an IP address.
- IP addresses come in three basic classful address types: Class A, Class B, and Class C.
- The Class A range of addresses has its first octet anywhere from 1 through 126. The standard Class A subnet mask is 255.0.0.0.
- A Class B address has its first octet anywhere from 128 through 191. Class B subnets use mask 255.255.0.0.
- Class C addresses range from 192 through 223, with the standard Class C subnet mask set to 255.255.255.0.
- Classless subnets do away with neat subnet masks. These subnet masks employ other binary representations in the masking process. For example, 255.255.255.0 is a standard Class C subnet mask, allowing for one subnet of 254 systems. Contrast that example with using subnet mask 255.255.255.240, which would allow for 14 subnets with 14 systems each.

Describe the functions of static and dynamic IP addresses

- Static addressing requires the IP address, subnet mask, and default gateway to be entered manually.

- Dynamic addressing uses either Dynamic Host Configuration Protocol (DHCP) or Bootstrap Protocol (BOOTP) to automatically assign an IP address, subnet mask, and default gateway to a network client.
- A network client is assigned an IP address from a DHCP server by exchanging the following packets: DHCP Discovery, DHCP Offer, DHCP Request, and DHCP Acknowledge.
- The data accepted by the DHCP client is called the DHCP lease, which is good for a fixed period of time. The time varies based on how the DHCP server was configured.
- A DHCP client that fails to acquire a DHCP lease from a DHCP server self-generates an IP address and subnet mask via Automatic Private IP Addressing (APIPA).
- The 127.0.0.1 loopback address used in testing is a reserved IP address.
- Private IP addresses include the following ranges: 10.0.0.0–10.255.255.255 (Class A), 172.16.0.0–172.31.255.255 (Class B), and 192.168.0.0–192.168.255.255 (Class C).

Key Terms

Address Resolution Protocol (ARP) *(147)*
Automatic Private IP Addressing (APIPA) *(162)*
Bootstrap Protocol (BOOTP) *(161)*
class license *(149)*
Classless Inter-Domain Routing (CIDR) *(150)*
default gateway *(144)*
DHCP lease *(162)*
dotted decimal notation *(140)*
dynamic addressing *(157)*
Dynamic Host Configuration Protocol (DHCP) *(161)*
host ID *(143)*
IFCONFIG *(143)*
Internet Assigned Numbers Authority (IANA) *(149)*
Internet Protocol version 4 (IPv4) *(137)*
IP addressing *(138)*
IPCONFIG *(142)*
loopback address *(164)*
multicast *(150)*
network ID *(143)*
protocol *(137)*
protocol suite *(137)*
routing table *(144)*
static addressing *(157)*
subnet mask *(146)*
subnetting *(150)*
unicast *(150)*

Key Term Quiz

Use the Key Terms list to complete the sentences that follow. Not all terms will be used.

1. The ________________ portion of an IP address resembles 150.0.0.0.
2. The ________________ portion of an IP address consists of from one to three octets, with the final octet between 1 and 254.
3. The single organization that distributes IP addresses is called the ________________.
4. The IP address 10.11.12.13 is a valid __________ address.
5. The command ________________ is a program that comes with Microsoft Windows to show TCP/IP settings.
6. The command ________________ is a program for UNIX/Linux/Macintosh used to show TCP/IP settings.
7. The ________________ protocol is used to translate MAC addresses to IP addresses.
8. Computers set for dynamic addressing that cannot locate a DHCP server use _____________ to assign themselves an IP address.
9. The ________________ is the IP address of the router.
10. The ________________ is a 32-bit binary number common to all computers on a network used to determine to which network a computer belongs.

Multiple-Choice Quiz

1. What is the result of converting 11110000.10111001.00001000.01100111 to dotted decimal notation?

A. 4.5.1.5

B. 240.185.8.103

C. 15.157.16.230

D. 103.8.185.240

2. What does IANA stand for?

A. International Association Numbers Authority

B. International Association Numbering Authority

C. Internet Assigned Numbering Authority

D. Internet Assigned Numbers Authority

3. Which of the following describe IPv4? (Select three.)

A. Uses decimal, not hexadecimal numbers

B. Uses periods, not colons, as separators

C. Uses four octets

D. Uses eight sets of characters

4. What is the result of converting 192.168.0.1 to binary?

A. 11000000.10101000.00000000.00000001

B. 11000000.10101000.00000000.10000000

C. 11000000.10101000.00000000.1

D. 11.10101.0.1

5. Which of the following are not valid IP addresses to assign to a Windows-based system? (Select two.)

A. 1.1.1.1/24

B. 127.0.0.1/24

C. 250.250.250.255/24

D. 192.168.0.1/24

6. Which of the following is a valid Class A IP address?

A. 22.33.44.55

B. 127.0.0.1

C. 250.250.250.250

D. 192.168.0.1

7. Which of the following is a valid Class B IP address?

A. 10.10.10.253

B. 191.254.254.254

C. 192.168.1.1

D. 223.250.250.1

8. Which of the following is a valid Class C IP address?

A. 50.50.50.50

B. 100.100.100.100

C. 192.168.0.254

D. 250.250.250.250

9. Which method sends a packet from a single computer to a group of interested computers?

A. Broadcast

B. Unicast

C. Multicast

D. Omnicast

10. What is the process used to take a single class of IP addresses and chop it up into multiple smaller groups? (Select two.)

A. CIDR

B. PING

C. Subnetting

D. Subnitting

11. Which statements about subnet masks are true? (Select two.)

A. Every network client has a unique subnet mask.

B. Every client on a network shares the same subnet mask.

C. A subnet mask consists of a string of zeroes followed by a string of ones.

D. A subnet mask consists of a string of ones followed by a string of zeroes.

12. In which order are packets created and sent when a client requests an IP address from a DHCP server?

A. DHCP discovery, DHCP offer, DHCP request, DHCP acknowledge

B. DHCP discovery, DHCP request, DHCP offer, DHCP acknowledge

C. DHCP request, DHCP offer, DHCP discovery, DHCP acknowledge

D. DHCP request, DHCP offer, DHCP acknowledge, DHCP discovery

13. Which of the following is *not* a valid classful subnet mask?

A. 255.0.0.0

B. 255.255.0.0

C. 255.255.255.0

D. 255.255.255.255

14. Which of the following is a valid classless subnet mask?

A. 255.255.255.240

B. 255.0.0.0

C. 255.255.0.0

D. 255.255.255.0

15. Which of the following IP addresses indicates a computer configured for dynamic addressing was unable to locate a DHCP server?

A. 255.255.255.255

B. 192.168.1.1

C. 127.0.0.1

D. 169.254.1.30

Essay Quiz

1. Use your Web browser to go to the www.webopedia.com Web site. Search for the full term TCP/IP. Write down its definition on a piece of paper, being sure to cite the exact Web site link to give credit to where you obtained the information.
2. You and a classmate are trying to calculate the number of possible IPv4 addresses versus IPv6 addresses. (The TCP/IP powers that be created the IPv6 addressing system to replace the IPv4 system discussed in this chapter. Because I feel IPv6 is going to be extremely important for all techs to understand in the future, this book devotes a full chapter to the subject—Chapter 13, "IPv6.") Research the Internet to discover exactly how many addresses are available for each of these numbering schemes. Document your findings in a short essay.
3. A new intern is confused about the CIDR notation for subnets, such as 192.168.1/24. In your own words, explain to him why the part in front of the slash represents only three of the four octets in an IP address and what the number after the slash is.

Lab Projects

• Lab Project 7.1

Use the Internet to research the components of what an individual TCP packet and an IP packet might look like. You can search on keywords "sample," "TCP," "IP," "session," and "packet." Create a reference document that has links to five sites with appropriate information. Save the document, so that the links contain hyperlinks that can be clicked on. Then write an additional paragraph describing your overall findings. Print one copy as well.

• Lab Project 7.2

Starting with the IP address 192.42.53.12, create a list of IP address ranges for six subnets.

• Lab Project 7.3

Log in to any available networked Windows 2000 or Windows XP computer. Click Start | Run, and type **cmd** to open a command prompt; from the command prompt, type `ipconfig /all`, and then press ENTER. Fill in as much information as you can from your screen onto a sheet like the following (or create one as directed by your instructor):

Host Name:
Primary DNS Suffix:
Node Type:
IP Routing Enabled:
WINS Proxy Enabled:
DNS Suffix Search List:
Connection-specific DNS Suffix:
Description:
Physical Address:
DHCP Enabled:
Autoconfiguration Enabled:
IP Address:
Subnet Mask:
Default Gateway:
DHCP Server:
DNS Servers:
Primary WINS Server:
Lease Obtained:
Lease Expires: