



Chapter

1

**Understanding
Windows Server 2003
Networking**



Microsoft has put an immense amount of time and effort into building Windows Server 2003. It's not fair to say that this operating system is an entirely new product because it still retains a great deal of core code from Windows 2000 and even Windows NT, Internet Information Server, and Exchange Server. Windows Server 2003 is a large, complicated, and very powerful operating system. To use it effectively, you have to understand how it works and how to make it do what you want it to do. This book is a study guide for the Implementing, Managing, and Maintaining a Microsoft Windows Server 2003 Network Infrastructure exam, so it makes sense to lead off with a discussion of the network protocols included in Windows Server 2003—what they're for, how they work, and what you can do with them.

Having a good frame of reference helps when comparing network protocols. To establish such a frame, this chapter will begin with the Open Systems Interconnection (OSI) network model, a sort of idealized way to stack various protocols together.

The OSI Model

The International Organization for Standardization (ISO) began developing the *Open Systems Interconnection (OSI)* reference model in 1977. It has since become the most widely accepted model for understanding network communication; once you understand how the *OSI model* works, you can use it to compare network implementations on different systems.

When you want to communicate with another person, you need to have two things in common: a communication language and a communication medium. Computer networks are no different; for communication to take place on a network composed of a variety of different network devices, both the language and medium must be clearly defined. The OSI model (and networking models developed by other organizations) attempts to define rules that cover both the generalities and specifics of networks:

- How network devices contact each other and, if they have different languages, how they communicate with each other
- Methods by which a device on a network knows when to transmit data and when not to
- Methods to ensure that network transmissions are received correctly and by the right recipient
- How the physical transmission media is arranged and connected
- How to ensure that network devices maintain a proper rate of data flow
- How bits are represented on the network media

The OSI model isn't a product. It's just a conceptual framework you can use to better understand the complex interactions taking place among the various devices on a network. It doesn't do anything in the communication process; appropriate software and hardware do the actual work. The OSI model simply defines which tasks need to be done and which protocols will handle those tasks at each of the seven layers of the model. The seven layers are as follows:

- Application (layer 7)
- Presentation (layer 6)
- Session (layer 5)
- Transport (layer 4)
- Network (layer 3)
- Data-Link (layer 2)
- Physical (layer 1)



You can remember the seven layers using a handy mnemonic, such as "All Pitchers Sometimes Take Naps During Preseason."

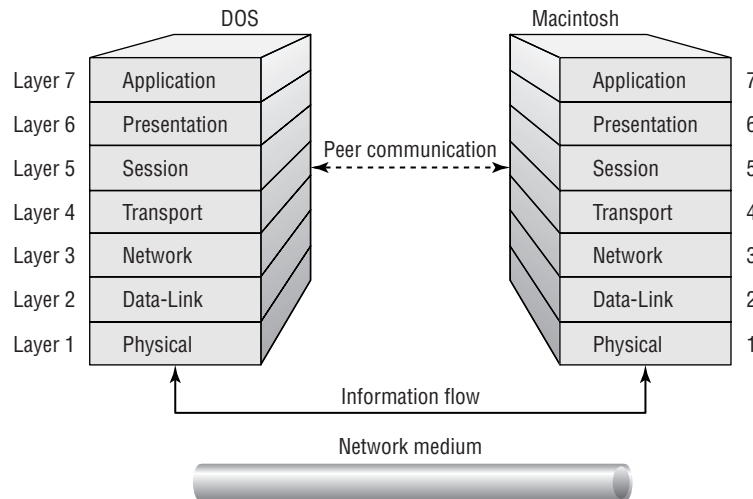
Each of the seven layers has a distinct function, which we'll explore a little later in the chapter.

Protocol Stacks

The OSI model splits communication tasks into smaller pieces called subtasks. Protocol implementations are computer processes that handle these subtasks. Specific protocols fulfill subtasks at specific layers of the OSI model. When these protocols are grouped together to complete a whole task, the assemblage of code is called a *protocol stack*. The stack is just a group of protocols, arranged in layers, that implements an entire communication process. Each layer of the OSI model has a different protocol associated with it. When more than one protocol is needed to complete a communication process, the protocols are grouped together in a stack. An example of a protocol stack is TCP/IP, which is widely used by Unix and the Internet—the TCP and IP protocols are implemented at different OSI layers.

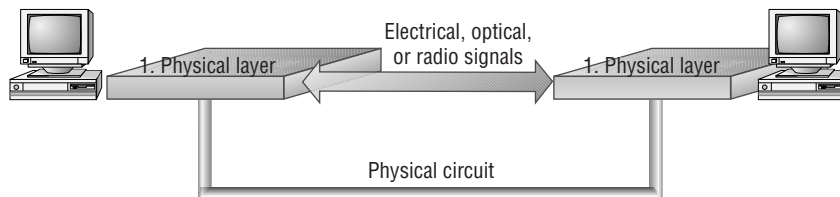
Each layer in the protocol stack receives services from the layer below it and provides services to the layer above it. It can be better explained like this: Layer N uses the services of the layer below it (layer N – 1) and provides services to the layer above it (layer N + 1).

For two computers to communicate, the same protocol stacks must be running on each computer. Each layer on both computers' stacks must use compatible protocols in order for the machines to communicate with each other. The computers can have different operating systems and still be able to communicate if they are running the same protocol stacks. For example, a DOS machine running TCP/IP can communicate with a Macintosh machine running TCP/IP (see Figure 1.1).

FIGURE 1.1 Each layer communicates with its counterparts on other network hosts.

The Physical Layer

The Physical layer is responsible for sending bits from one computer to another. Physical layer components don't care what the bits *mean*; their job is to get the bits from point A to point B, using whatever kind of optical, electrical, or wireless connection that connects the points. This level defines physical and electrical details, such as what will represent a 1 or a 0, how many pins a network connector will have, how data will be synchronized, and when the network adapter may or may not transmit the data (see Figure 1.2).

FIGURE 1.2 The Physical layer makes a physical circuit with electrical, optical, or radio signals.

The Physical layer addresses all the minutiae of the actual physical connection between the computer and the network medium, including the following:

- Network connection types, including multipoint and point-to-point connections.
- Physical topologies, or how the network is physically laid out (e.g., bus, star, or ring topologies).
- Which analog and digital signaling methods are used to encode data in the analog and digital signals.

- Bit synchronization, which deals with keeping the sender and receiver in synch as they read and write data.
- Multiplexing, or the process of combining several data channels into one.
- Termination, which prevents signals from reflecting back through the cable and causing signal and packets errors. It also indicates the last node in a network segment.

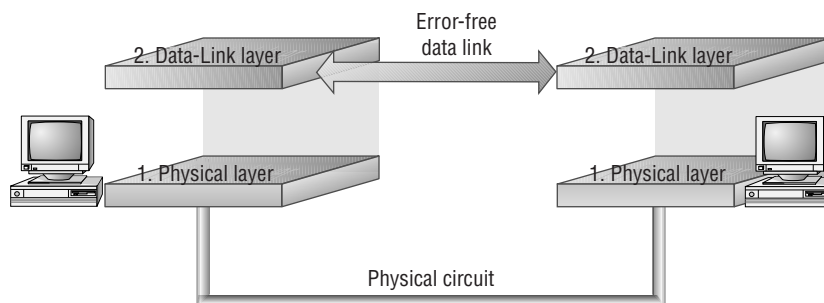
The Data-Link Layer

The Data-Link layer provides for the flow of data over a single physical link from one device to another. It accepts packets from the Network layer and packages the information into data units called frames; these frames are presented to the Physical layer for transmission. The Data-Link layer adds control information, such as frame type, to the data being sent.

This layer also provides for the error-free transfer of frames from one computer to another. A *cyclic redundancy check (CRC)* added to the data frame can detect damaged frames, and the Data-Link layer in the receiving computer can request that the CRC information be present so that it can check incoming frames for errors. The Data-Link layer can also detect when frames are lost and request that those frames be sent again.

In broadcast networks such as Ethernet, all devices on the LAN receive the data that any device transmits. (Whether a network is broadcast or point-to-point is determined by the network protocols used to transmit data over it.) The Data-Link layer on a particular device is responsible for recognizing frames addressed to that device and throwing the rest away, much as you might sort through your daily mail to separate good stuff from junk. Figure 1.3 shows how the Data-Link layer establishes an error-free connection between two devices.

FIGURE 1.3 The Data-Link layer establishes an error-free link between two devices.



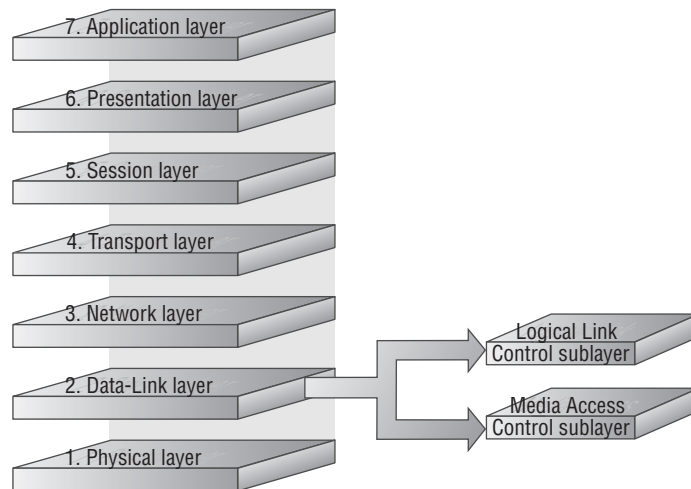
The Institute of Electrical and Electronics Engineers (IEEE) developed a protocol specification known as IEEE 802.X. (802.2 is the standard that divides this layer into two sublayers. The MAC layer varies for different network types and is described further in standards 802.3 through 802.5.) As part of that specification (which today we know as Ethernet), the Data-Link layer is split into two sublayers:

- The *Logical Link Control (LLC)* layer establishes and maintains the logical communication links between the communicating devices.

- The *Media Access Control (MAC)* layer acts like an airport control tower—it controls the way multiple devices share the same media channel in the same way that a control tower regulates the flow of air traffic into and out of an airport.

Figure 1.4 illustrates the division of the Data-Link layer into the LLC and MAC layers.

FIGURE 1.4 The IEEE split the ISO Data-Link layer into the LLC sublayer and the MAC sublayer.



The LLC sublayer provides *Service Access Points (SAPs)* that other computers can refer to and use to transfer information from the LLC sublayer to the upper OSI layers. This is defined in the 802.2 standard.

The MAC sublayer, the lower of the two sublayers, provides for shared access to the network adapter and communicates directly with network interface cards. Network interface cards have a unique 12-digit hexadecimal MAC address (frequently called the hardware Ethernet address) assigned before they leave the factory where they are made. The LLC sublayer uses MAC addresses to establish logical links between devices on the same LAN.

The Network Layer

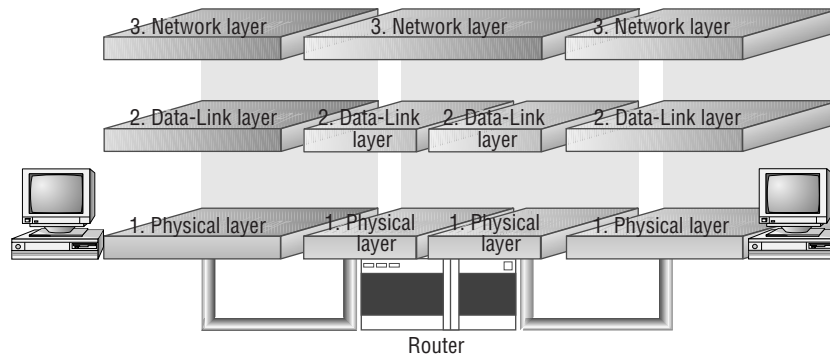
The Network layer handles moving packets between devices that are more than one link away from each other. It makes routing decisions and forwards packets as necessary to help them travel to their intended destination. In larger networks, there may be intermediate devices and subnetworks between any two end systems. The network layer makes it possible for the Transport layer (and layers above it) to send packets without being concerned with whether the end system is on the same piece of network cable or on the other end of a large wide area network.

To do its job, the Network layer translates logical network addresses into physical machine addresses (MAC addresses, which operate at the Data-Link layer). The Network layer also determines the quality of service (such as the priority of the message) and the route a message will take if there are several ways a message can get to its destination.

The Network layer also may split large packets into smaller chunks if the packet is larger than the largest data frame the Data-Link layer will accept. The network reassembles the chunks into packets at the receiving end.

Intermediate systems that perform only routing and relaying functions and do not provide an environment for executing user programs can implement just the first three OSI network layers. Figure 1.5 shows how the Network layer moves packets across multiple links in a network.

FIGURE 1.5 The Network layer moves packets across links to their destination.



The Network layer performs several important functions that enable data to arrive at its destination. The protocols at this layer may choose a specific route through an internetwork to avoid the excess traffic caused by sending data over networks and segments that don't need access to it. The Network layer serves to support communications between logically separate networks. This layer is concerned with the following:

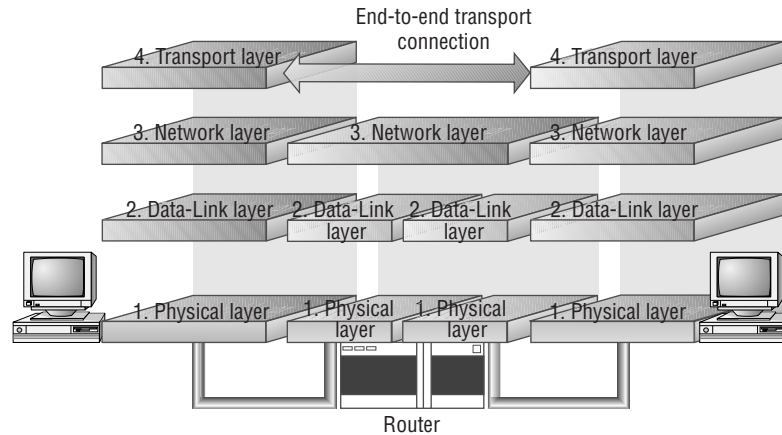
- Addressing, including logical network addresses and services addresses
- Circuit, message, and packet switching
- Route discovery and route selection
- Connection services, including Network layer flow control, Network layer error control, and packet sequence control
- Gateway services

In Windows Server 2003, the various routing services for TCP/IP, AppleTalk, and Internetwork Packet Exchange/Sequenced Packet Exchange (IPX/SPX) perform Network layer services (see Chapter 9, “Managing IP Routing,” for more on these services). In addition, the TCP/IP, AppleTalk, and IPX stacks provide routing capacity for those protocols.

The Transport Layer

The Transport layer ensures that data is delivered error free, in sequence, and with no losses or duplications. This layer also breaks large messages from the Session layer into smaller packets to be sent to the destination computer and reassembles packets into messages to be presented to the Network layer. The Transport layer typically sends an acknowledgment to the originator for messages received (as in Figure 1.6).

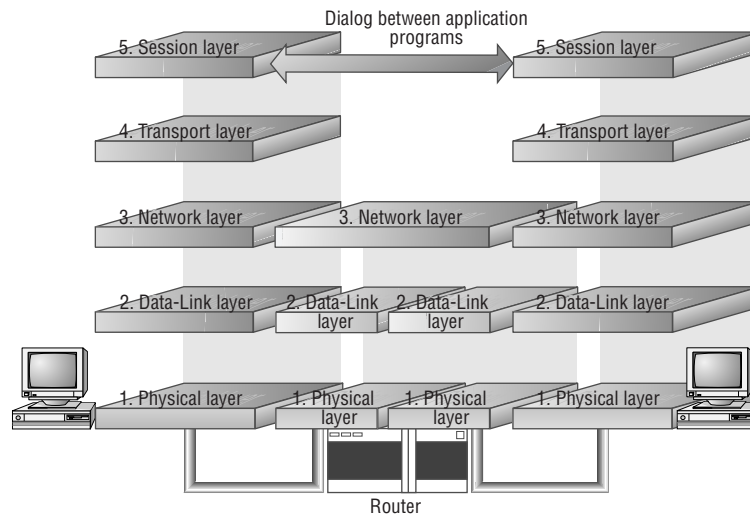
FIGURE 1.6 The Transport layer provides end-to-end communication with integrity and performance guarantees.



The Session Layer

The Session layer allows applications on separate computers to share a connection called a session. This layer provides services, such as name lookup and security, that allow two programs to find each other and establish the communication link. The Session layer also provides for data synchronization and checkpointing so that in the event of a network failure, only the data sent after the point of failure would need to be resent. This layer also controls the dialog between two processes and determines who can transmit and who can receive at what point during the communication (see Figure 1.7).

FIGURE 1.7 The Session layer allows applications to establish communication sessions with each other.

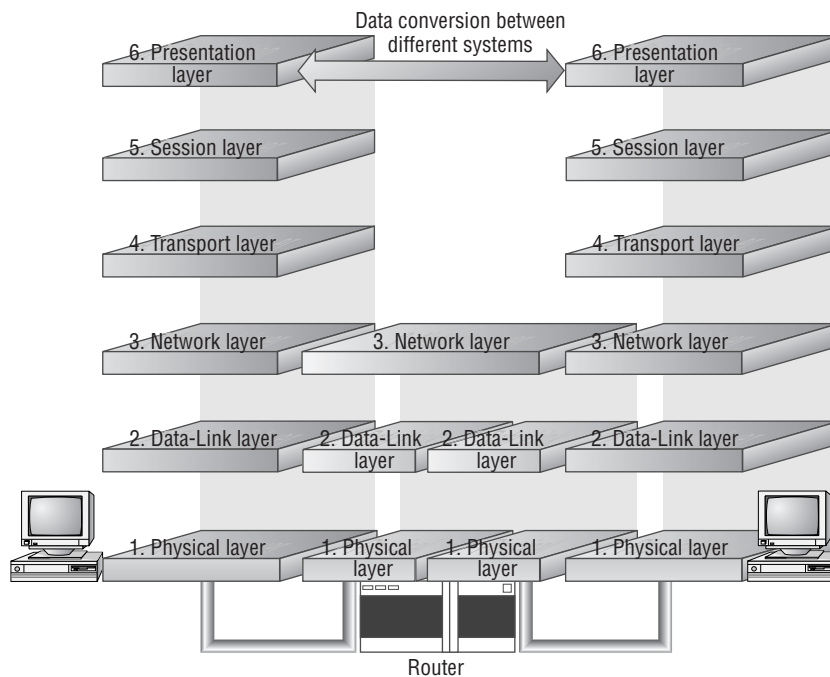


The Presentation Layer

The Presentation layer translates data between the formats the network requires and the formats the computer expects. The Presentation layer performs protocol conversion; data translation, compression, and encryption; character set conversion; and the interpretation of graphics commands.

The network redirector, long a part of Windows networking, operates at this level. The redirector is what makes the files on a file server visible to the client computer. The network redirector also makes remote printers act as though they are attached to the local computer. Figure 1.8 shows the Presentation layer's role in the protocol stack.

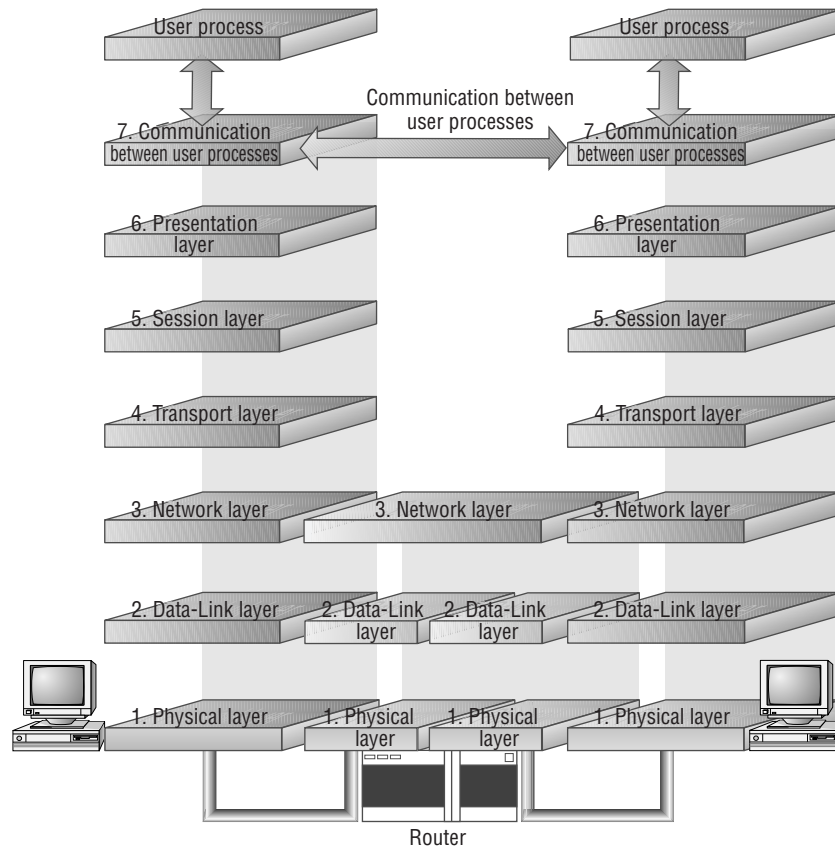
FIGURE 1.8 The Presentation layer allows applications to establish communication sessions with each other.



The Application Layer

The Application layer is the topmost layer of the OSI model, and it provides services that directly support user applications, such as database access, e-mail, and file transfers. It also allows applications to communicate with applications on other computers as though they were on the same computer. When a programmer writes an application program that uses network services, this is the layer the application program will access. For example, Internet Explorer uses the Application layer to make its requests for files and web pages; the Application layer then passes those requests down the stack, with each succeeding layer doing its job (as in Figure 1.9).

FIGURE 1.9 The Application layer is where the applications function, using lower levels to get their work done.



Communication between Stacks

When a message is sent from one machine to another, it travels down the layers on one machine and then up the layers on the other machine, as shown in Figure 1.10.

As the message travels down the first stack, each layer it passes through (except the Physical layer) adds a header. These headers contain pieces of control information that are read and processed by the corresponding layer on the receiving stack. As the message travels up the stack of the other machine, each layer removes the header added by its peer layer and uses the information it finds to figure out what to do with the message contents (see Figure 1.11).

As an example, consider the network we're using while writing this book. It's a TCP/IP network containing several Windows 2000, Windows Server 2003, Macintosh, and Windows NT machines, all connected using the TCP/IP protocol. When we mount a share from our Windows Server 2003 file server on the Mac desktop, at layer 7, the Mac Finder requests something from the Windows Server 2003. This request is sent to the Mac's layer 6, which receives the request as a data packet, adds its own header, and passes the packet down to layer 5. At layer 5,

the process is repeated, and it continues until the packet makes it to the Physical layer. The physical layer is responsible for actually moving the bits across the network wiring in the office, so it carries the request packet to a place where the Windows Server 2003 machine can “hear” it. At that point, the request packet begins its journey up the layers on the Windows Server 2003 file server. The header that was put on at the Data-Link layer of the Mac OS is stripped off at the Data-Link layer on the Windows Server 2003 machine. The Windows Data-Link layer driver performs the tasks requested in the header and passes the requests to the next, higher layer. This process is repeated until the Windows Server 2003 file server receives the packet and interprets the request. The Windows Server 2003 would then formulate an appropriate response and send it to the Mac.

FIGURE 1.10 Traffic flows down through the stack on one computer and up the stack on the other.

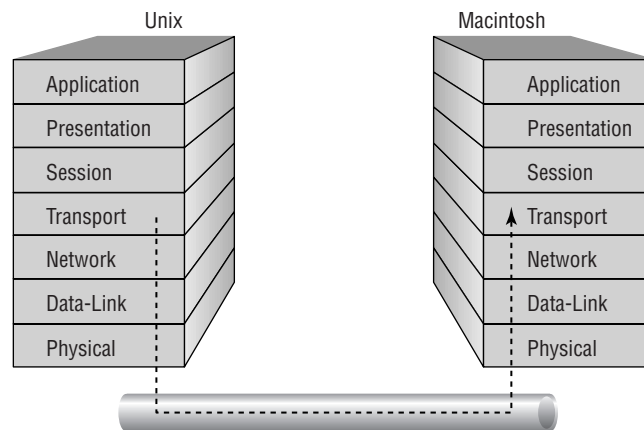
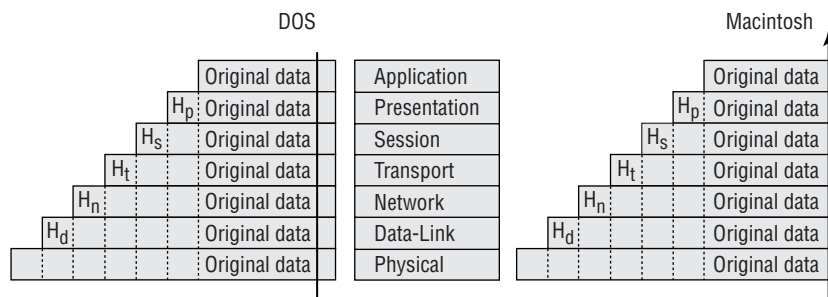


FIGURE 1.11 As packets flow up and down the stacks, each layer adds or removes necessary control information.



H_p = Presentation header
 H_s = Session header
 H_t = Transport header
 H_n = Network header
 H_d = Data-Link header

Microsoft's Network Components and the OSI Model

Because the OSI model is so abstract, it can be hard to tell how its concepts relate to the actual network software and hardware you use in the real world. The following sections will make the link clearer. We will introduce you to the specific protocols that are included with Windows Server 2003 and see how they apply to the various layers of the OSI model.

Device Drivers and the OSI Model

Every hardware device in a computer requires a software-based device driver to make it work. Some drivers—for instance, the driver for an integrated device electronics (IDE) hard disk or for the keyboard—are built into the operating system. Other devices require that drivers be installed separately when the device is attached or installed in the computer. Windows Server 2003 really blurs this distinction because it includes drivers for several hundred different network cards but if your card isn't on the list, you will need to install drivers provided by the manufacturer.

In the past (e.g., when Windows 3.11 was introduced), network drivers were vendor specific, for both the operating system and the card. You might, for instance, have a difficult time if you wanted to put a 3Com Ethernet card and an IBM Token Ring card in the same server. Worse yet, most drivers could only be bound to a single protocol stack and a single card, so you couldn't have two cards using TCP/IP on one server.

A variety of vendors tried to solve this problem by developing driver interfaces that allowed multiple cards to be bound to multiple protocols. Apple and Novell developed the *Open Data-link Interface (ODI)*, and Microsoft countered with the *Network Driver Interface Specification (NDIS)*. Microsoft's operating systems have supported NDIS ever since, making it possible to bind either multiple protocols to one card or the same protocol to multiple cards.

Network adapter cards and drivers provide the services corresponding to the Data-Link layer in the OSI model. In the IEEE model, the Data-Link layer is split into the Logical Link Control (LLC) sublayer—which corresponds to the software drivers—and the Media Access Control (MAC) sublayer—which corresponds to the network adapter. You can think of the drivers as intermediaries between the higher layers and the card hardware that handles the business of forming packets and stuffing them into a wire.

The Basics of Network Protocols

Protocols are nothing more than an agreed-upon way in which two objects (people, computers, home appliances, etc.) can exchange information. There are protocols at various levels in the OSI model. In fact, it is the protocols at a particular level in the OSI model that provide that level's functionality. Protocols that work together to provide a layer or layers of the OSI model are known as a protocol stack or protocol suite. The following sections explain how network protocols move data between machines.

How Protocols Work

A protocol is a set of basic steps that both computers must perform in the right order. For instance, for one computer to send a message to another computer, the first computer must perform the steps given in the following general example:

1. Break the data into small sections called packets.
2. Add addressing information to the packets, identifying the destination computer.
3. Deliver the data to the network card for transmission over the network.

The receiving computer must perform these steps:

1. Accept the data from the network adapter card.
2. Remove the transmitting information that was added by the transmitting computer.
3. Reassemble the packets of data into the original message.

Each computer needs to perform the same steps, in the same way and in the correct order, so that the data will arrive and be reassembled correctly. If one computer uses a protocol with different steps or even the same steps with different parameters (such as different sequencing, timing, or error correction), the two computers won't be able to communicate with each other.

Network Packets

Networks primarily send and receive small chunks of data called *packets*. Network protocols construct, modify, and disassemble packets as they move data down the sending stack, across the network, and back up the OSI stack of the receiving computer. Packets have the following components:

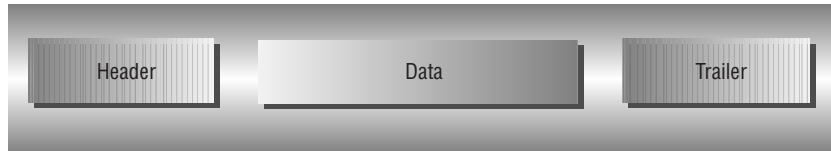
- A source address specifying the sending computer
- A destination address specifying where the packet is being sent
- Instructions that tell the computer how to pass the data along
- Reassembly information (if the packet is part of a longer message)
- The data to be transmitted to the remote computer (often called the *packet payload*)
- Error-checking information to ensure that the data arrives intact

These components are assembled into slightly larger chunks; each packet contains three distinct parts (listed here and seen in Figure 1.12), and each part contains some of the components listed previously:

Header A typical header includes an alert signal to indicate that the data is being transmitted, source and destination addresses, and clock information to synchronize the transmission.

Data This is the actual data being sent. It can vary (depending on the network type) from 48 bytes to 4 kilobytes.

Trailer The contents of the trailer (or even the existence of a trailer) vary among network types, but it typically includes a CRC. The CRC helps the network determine whether or not a packet has been damaged in transmission.

FIGURE 1.12 A packet consists of a header, the data, and a trailer.

Protocols and Binding

Many different protocol stacks can perform network functions, and many different types of network interface cards can be installed in a computer. A computer may have more than one card, and a computer may use more than one protocol stack at the same time.

The *binding* process is what links the protocol stack to the network device driver for the network interface adapter. Several protocols can be bound to the same card; for instance, both TCP/IP and AppleTalk can be bound to the same Ethernet adapter. In addition, one computer with several interface adapters—for instance, a server that must be able to communicate with both a local area network and a network backbone—can have the same protocol bound to two or more network cards.

The binding process can be used throughout the OSI layers to link one protocol stack to another. The device driver (which implements the Data-Link layer) is bound to the network interface card (which implements the Physical layer). TCP/IP can be bound to the device driver, and the NWLINK Session layer can be bound to the device driver.

Bindings are particularly important to Windows Server 2003 because you'll often want to change the bindings so that protocols you don't need on a particular network aren't bound to some network adapters. For example, it's very common to unbind the NWLINK protocol from the network card connected to a web server's Internet connection.

Determining Connections

There are two ways that communication between computers can be arranged: using connectionless protocols and using connection-oriented protocols. It's important to understand the differences between them because different Windows Server 2003 services use both types.

Connectionless Protocols

It might seem odd to talk about a connectionless protocol for networks, but you use at least two of them just about every day: radio and television. Connectionless systems assume that all data will get through, so the protocol doesn't guarantee delivery or correct packet ordering. Think of shouting a message out of your window to someone walking by outside—there's no guarantee that they'll hear you, but it's quick and easy. These optimistic assumptions mean that there's no protocol overhead spent on these activities, so connectionless protocols tend to be fast. The *User Datagram Protocol (UDP)*, which is part of the TCP/IP protocol standard, is an example of a connectionless Internet transport protocol.

Connectionless systems normally work pretty well on lightly loaded networks like most local area networks. Unfortunately, they break down quickly in large or heavily loaded networks where packets can be dropped due to line noise or router congestion.

All is not lost for connectionless transports, however, because higher-level protocols will know what data has not reached its destination after some time and request a retransmission. However, connectionless systems don't necessarily return data in sequential order, so the higher-level protocol must sort out the data packets.

Connection-Oriented Protocols

Connection-oriented systems work more like your telephone—you have to dial a number and establish a connection to the other end before you can send a message. Connection-oriented protocols pessimistically assume that some data will be lost or disordered in most transmissions. They guarantee that transmitted data will reach its destination in the proper sequence and that all data will get through. To accomplish this, connection-oriented protocols retain the transmitted data and negotiate for a retransmission when needed. Once all the needed data has arrived at the remote end, it can be reassembled into its proper sequence and passed to the higher-level protocols. This means that any application can depend on a connection-oriented transport to reliably deliver data exactly as it was transmitted. Transmission Control Protocol (TCP) is an example of a connection-oriented Internet protocol.

For local area systems where data isn't likely to be dropped, it makes sense to push serialization and guaranteed delivery up to higher-level protocols that are less efficient because they won't be used often anyway. But in wide area networks like the Internet, it would simply take too much time for higher-level protocols to sort out what data had been sent and what was missing, so the transport protocol simply takes measures to guarantee that all the data gets through in order.

Network Protocols and Windows Server 2003

So far, you've been reading a lot of abstract material that might not seem pertinent to Windows Server 2003. Now we'll show you the network protocols included with Windows Server 2003 and how each fits into the models you've read about up to this point.

There are a number of protocol stacks used in the world's networks today. Besides NetWare, AppleTalk, NetBIOS, and TCP/IP, there are a bunch of specialty protocols like IBM's Systems Network Architecture (SNA), Digital's (now HP/Compaq's) DECnet, and others. Even though these protocols actually work at different levels of the OSI model, they fall neatly into three distinct groups, as seen in the following list and in Figure 1.13:

- Application protocols provide for application-to-application interaction and data exchange.
- Transport protocols establish communication sessions between computers.
- Network protocols handle issues such as routing and addressing information, error checking, and retransmission requests.

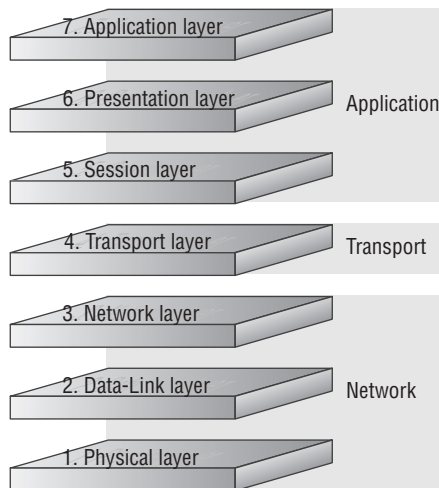
Microsoft networking products come with three network transports—NWLink IPX/SPX, AppleTalk, and TCP/IP—and each is intended for networks of different sizes with different requirements. Each network transport has different strengths and weaknesses. *NWLink* is intended for medium-sized networks (in a single facility, perhaps) or for networks that require access to Novell NetWare file servers. AppleTalk's primary use is interoperating with Macintosh computers (a topic that's too specialized to discuss further here). *TCP/IP* is a complex transport sufficient for globe-spanning networks such as the Internet, and Microsoft is doing

everything possible to position TCP/IP as a one-size-fits-all network protocol. TCP/IP is required to use Active Directory and is the default protocol for Windows Server 2003.



Windows Server 2003 does not include support for the now defunct NetBEUI protocol. NetBEUI was not routable and Microsoft deemed the protocol unsuitable for enterprise-scale networks.

FIGURE 1.13 The OSI protocol stack can be simplified by grouping its layers into three new categories.



NWLink

NWLink IPX/SPX is Microsoft's implementation of Novell's IPX/SPX protocol stack, which is used in Novell NetWare. In fact, it's fair to say that NWLink IPX/SPX is nothing more than IPX for Windows.

NWLink IPX/SPX is included with Windows Server 2003 primarily to allow Windows Server 2003 to interconnect with legacy Novell NetWare servers and clients. Microsoft clients and servers can then be added to existing network installations, over time easing the migration between platforms and obviating the need for a complete cutover from one networking standard to another.

The advantages of NWLink IPX/SPX include the following:

- It's easy to set up and manage.
- It's routable.
- It's easy to connect to installed NetWare servers and clients.

However, NWLink IPX/SPX has some disadvantages, such as these:

- With NWLink IPX/SPX, it is difficult to exchange traffic with other organizations.

- It has limited support in Windows Server 2003.
- It doesn't support standard network management protocols.

Truly large networks (networks that connect many organizations) may find that NWLink IPX/SPX is difficult to use because there is no effective central IPX addressing scheme—as there is with TCP/IP—to ensure that two networks don't use the same address numbers. IPX doesn't support the wide range of network management tools available for TCP/IP.

TCP/IP

TCP/IP is actually two sets of protocols bundled together: The *Transmission Control Protocol (TCP)* and the *Internet Protocol (IP)*. TCP/IP, and a suite of related protocols, were developed by the Department of Defense's Advanced Research Projects Agency (ARPA, or later DARPA) beginning in 1969. The original goal was to develop network protocols that were robust enough to route communications around damage caused by nuclear war. That design goal was never tested, but some aspects of that design have led to the redundant, distributed whole we call the Internet.

TCP/IP is by far the most widely used protocol for interconnecting computers, and it is the protocol of the Internet. This is because, although ARPA originally created TCP/IP to connect military networks together, it provided the protocol standards to government agencies and universities free of charge. The academic world leapt at the chance to use a robust protocol to interconnect their networks, and the Internet was born. Many organizations and individuals collaborated to create higher-level protocols for everything from newsgroups, mail transfer, and file transfer to printing, remote booting, and even document browsing.

TCP/IP is currently the protocol of choice for most networks because of its rapid and widespread adoption. TCP/IP is used for networks that span more than one metropolitan area or to connect to (or over) the Internet.

TCP/IP has some significant advantages:

- Broad connectivity among all types of computers and servers, including direct access to the Internet
- Strong support for routing, using a number of flexible routing protocols (see Chapter 9 for more on these protocols)
- Support for advanced name and address resolution services (which will be covered in more depth later in this book): the Domain Name Service (DNS), the Dynamic Host Configuration Protocol (DHCP), and the Windows Internet Name Service (WINS)
- Support for a wide variety of Internet-standard protocols, including protocols for mail transport, web browsing, and file and print services
- Centralized network number and name assignment, which facilitates internetworking between organizations

TCP/IP has some disadvantages:

- It's harder to set up than IPX.
- Its routing and connectivity features impose relatively high overhead.
- It's slower than IPX.

Even given these disadvantages, it's the core protocol that Windows Server 2003 depends on for all its network services. In fact, most of this book focuses on TCP/IP and related services.

TCP (and UDP) relies on port numbers assigned by the Internet Assigned Numbers Authority (IANA) to forward packets to the appropriate application process. Port numbers are 16-bit integers that are part of a message header and identify the process that the packet should be associated with.

For example, let's say that a client has a copy of Internet Explorer and a copy of Outlook Express open at the same time. Both applications are sending TCP requests across the Internet to retrieve web pages and e-mail, respectively. How does the computer know which packets to forward to Internet Explorer and which packets to forward to Outlook Express? Every packet destined for Internet Explorer has a port number of 80 in the header, and every packet destined for Outlook Express has a port number of 110 in the header. Table 1.1 contains a list of the major port numbers you might need to know for the exam. You can also visit www.iana.org to get the most current full list of port numbers.

TABLE 1.1 Well-known Port Numbers

Port Number	Description
20	File Transfer Protocol (FTP) data
21	File Transfer Protocol (FTP) control
23	Telnet
25	Simple Mail Transfer Protocol (SMTP)
80	Hypertext Transfer Protocol (HTTP), Web
88	Kerberos
110	Post Office Protocol v3 (POP3)
443	Secure HTTP (HTTPS)



Real World Scenario

Understanding the OSI Model and Troubleshooting

The company you work for has several regional offices spread around the country. Your job is to make sure the resources on the Windows Server 2003 network, which include manufacturing, inventory, and sales information, are available at all times. If the sales information from the regional offices isn't collected and updated to the manufacturing and inventory programs, the company won't be able to supply its customers effectively. The users of the network aren't particularly interested in the technical nuts and bolts of the system, but they do care when the system is down.

At the same time, you're studying for your MSCE and wondering how the abstract notions of the OSI model are relevant to your job. A support call comes in from a user who can't connect to a printer on a Windows Server 2003 machine in another region where an executive management meeting is taking place. The user is down the hall from you, so you drop everything and run down to take a look.

With the OSI model fresh in mind, you approach the problem in terms of layers of functionality. You ping the address of your router, and it comes back fine. You now know that the Physical, Data-Link, Network, and Transport layers are working fine, which means that you have eliminated cable and basic protocol problems. Your browser also seems to work fine because you can reach random sites. When you ping the name of the Windows Server 2003 machine that hosts the printer, you get the "request timed out" message. But when you ping the IP address directly, the reply shows a healthy connection, implying that you have a name resolution problem. You connect to the printer using a Net Use with the IP address and begin the task of looking at your WINS server.

By breaking down your troubleshooting tactics into the general OSI layers, you can get a better gauge of where the problem lies and which services to look at, depending on where in the OSI model the symptoms appear. Although the OSI model is fairly abstract, when it's applied appropriately, it gives you a structure for thinking about your overall network and provides a framework for following methodical troubleshooting tactics.

Understanding IP Addressing

Understanding IP addressing is critical to understanding how TCP/IP works. An IP address is a numeric identifier assigned to each machine on a TCP/IP network. It designates the location of the device it is assigned to on the network. This type of address is a software address, not a hardware address, which is hard-coded in the machine or network interface card.

In the following sections, you will see how IP addresses are used to uniquely identify every machine on a network.



We're going to assume you're comfortable with binary notation and math for the remainder of this section.

The Hierarchical IP Addressing Scheme

An IP address is made up of 32 bits of information. These bits are divided into four sections (sometimes called octets or quads) containing 1 byte (8 bits) each. There are three methods for specifying an IP address:

- Dotted-decimal, as in 130.57.30.56

- Binary, as in 10000010.00111001.00011110.00111000
- Hexadecimal, as in 82 39 1E 38

All of these examples represent the same IP address.

The 32-bit IP address is a structured address, or *hierarchical address*, as opposed to a flat address, or nonhierarchical one. Although IP could have used either flat or hierarchical addressing, its designers chose hierarchical addressing—for a very good reason, as you will see.

What's the difference between these two types of addressing? A good example of a flat addressing scheme is a driver's license number. There's no partitioning to it; the range of legal numbers isn't broken up in any meaningful way (say, by county of residence or date of issuance). If this method had been used for IP addressing, every machine on the Internet would have needed a totally unique address, just as each driver's license number is unique. The good news about flat addressing is that it can handle a large number of addresses, namely 4.3 billion (a 32-bit address space with two possible values for each position—either 0 (zero) or 1 (one)—giving you 2^{32} , which equals approximately 4.3 billion). The bad news—and the reason why flat addressing isn't used in IP—relates to routing. If every address were totally unique, every router on the Internet would need to store the address of each and every *other* machine on the Internet. It would be fair to say that this would make efficient routing impossible, even if only a fraction of the possible addresses were used.

The solution to this dilemma is to use a hierarchical addressing scheme that breaks the address space up into ordered chunks. Telephone numbers are a great example of this type of addressing. The first section of a telephone number, the area code, designates a very large area; the area code is followed by the prefix, which narrows the scope to a local calling area. The final segment, the customer number, zooms in on the specific connection. By looking at a number like 256-233-xxxx, you can quickly determine that the number is located in the northern part of Alabama (area code 256) in the Athens/East Limestone area (the 233 exchange).

IP addressing works the same way. Instead of treating the entire 32 bits as a unique identifier, one part of the IP address is designated as the *network address* and the other part as a *node address*, giving it a layered, hierarchical structure.

The network address uniquely identifies each network. Every machine on the same network shares that network address as part of its IP address. In the IP address 130.57.30.56, for example, 130.57 is the network address.

The node address is assigned to, and uniquely identifies, each machine in a network. This part of the address must be unique because it identifies a particular machine—an individual as opposed to a network that is a group. This number can also be referred to as a host address. In the sample IP address 130.57.30.56, .30.56 is the node address.

The designers of the Internet decided to create classes of networks based on network size. For the small number of networks possessing a very large number of nodes, they created the Class A network. At the other extreme is the Class C network, reserved for the numerous networks with a small number of nodes. The class distinction for networks in between very large and very small is predictably called a Class B network. How you subdivide an IP address into a network and node address is determined by the class designation of your network. Table 1.2 provides a summary of the three classes of networks, which will be described in more detail in the following sections.

TABLE 1.2 Network Address Classes

Class	Format	Leading Bit Pattern	Decimal Range of First Byte of Network Address	Maximum Number of Networks	Maximum Nodes per Network
A	Node	0	1–126	126	16,777,214
B	Node	10	128–191	16,384	65,534
C	Node	110	192–223	2,097,152	254

To ensure efficient routing, Internet designers defined a mandate for the leading bits section of the address for each different network class. For example, because a router knows that a Class A network address always starts with a 0, the router might be able to speed a packet on its way after reading only the first bit of its address. Table 1.2 illustrates how the leading bits of a network address are defined.

Some IP addresses are reserved for special purposes and shouldn't be assigned to nodes by network administrators. Table 1.3 lists the reserved IP addresses.

TABLE 1.3 Special Network Addresses

Address	Function
Network portion of the address set to all 0s (zeros)	This network or subnet (i.e., the network or subnet that you are currently a part of)
Network portion of the address set to all 1s	This network and all related subnets
Network address 127	Reserved for loopback tests. Designates the local node and allows that node to send a test packet to itself without generating network traffic.
Node address of all 0s	Used when referencing a network without referring to any specific nodes on that network. Usually used in routing tables.
Node address of all 1s	Broadcast address for all nodes on the specified network; for example, 128.2.255.255 means all nodes on network 128.2 (Class B address).
Entire IP address set to all 0s	Used by RIP to designate the default route.
Entire IP address set to all 1s (same as 255.255.255.255)	Broadcast to all nodes on the current network; sometimes called an "all 1s broadcast."

In the following sections, we will look at the three different network types.

Class A Networks

In a Class A network, the first byte is the network address, and the three remaining bytes are used for the node addresses. The Class A format is **Network.Node.Node.Node**.

For example, in the IP address 49.22.102.70, 49 is the network address and 22.102.70 is the node address. Every machine on this particular network would have the distinctive network address of 49; within that network, though, you could have a large number of machines.

The length of a Class A network address is a byte, and the first bit of that byte is reserved, so 7 bits in the first byte remain for manipulation. That means that the maximum number of Class A networks that could be created is 128. Why? Each of the seven bit positions can be either a 0 or a 1; this gives you a total of 2^7 positions: a total of 128. To complicate things further, it was also decided that the network address of all 0s (0000 0000) would be reserved. This means the actual number of usable Class A network addresses is 128 minus 1, or 127. There's actually another reserved Class A address too—127—which consists of a network address of all 1s (111 1111). Because you start with 128 addresses and two are reserved, you're left with 126 possible Class A network addresses.

Each Class A network has 3 bytes (24 bit positions) for the node address of a machine, which means that there are 2^{24} —or 16,777,216—unique combinations. Therefore, there are precisely that many possible unique node addresses for each Class A network. Because addresses with the two patterns of all 0s and all 1s are reserved, the actual maximum usable number of nodes for a Class A network is 2^{24} minus 2, which equals 16,777,214.

Class B Networks

In a Class B network, the first 2 bytes are assigned to the network address, and the remaining 2 bytes are used for node addresses. The format is **Network.Network.Node.Node**.

For example, in the IP address 130.57.30.56, the network address is 130.57 and the node address is 30.56.

The network address is 2 bytes, so there would be 2^{16} unique combinations. But the Internet designers decided that all Class B networks should start with the binary digits 1 and 0. This leaves 14 bit positions to manipulate; therefore, there are 16,384 (or 2^{14}) unique Class B networks.

This gives you an easy way to recognize Class B addresses. If the first 2 bits of the first byte can only be 10, that gives us a decimal range from 128 up to 191. Remember that you can always easily recognize a Class B network by looking at its first byte—even though there are 16,384 different Class B networks. If the first number in the address falls between 128 and 191, it is a Class B network.

A Class B network has 2 bytes to use for node addresses. This is 2^{16} minus the two patterns in the reserved-exclusive club (all 0s and all 1s), for a total of 65,534 possible node addresses for each Class B network.

Class C Networks

The first 3 bytes of a Class C network are dedicated to the network portion of the address, with only 1 byte remaining for the node address. The format is **Network.Network.Network.Node**.

In the example IP address 198.21.74.102, the network address is 198.21.74 and the node address is 102.

In a Class C network, the first three bit positions are always binary 110. The calculation is such: 3 bytes, or 24 bits, minus 3 reserved positions leaves 21 positions. There are therefore 2^{21} or 2,097,152 possible Class C networks, each of which has 254 possible node addresses (remember, all 0s and all 1s are special addresses: $256 - 2 = 254$).

The lead bit pattern of 110 equates to decimal 192 and runs through 223. Remembering our handy easy-recognition method, this means that (although there are a total of 2,097,152 possible Class C networks) you can always spot a Class C address if the first byte is between 192 and 223.

Each unique Class C network has 1 byte to use for node addresses. This leads to 2^8 , or 256, minus the two special club patterns of all 0s and all 1s, for a total of 254 node addresses for each Class C network.



Class D networks, used for multicasting only, use the address range 224.0.0.0 to 239.255.255.255. Class E networks (reserved for future use at this point) cover 240.0.0.0 to 255.255.255.255.

Subnetting a Network

If an organization is large and has lots of computers, or if its computers are geographically dispersed, it makes good sense to divide its colossal network into smaller ones connected by routers. These smaller nets are called *subnets*. The benefits to using subnets include the following:

Reduced network traffic We all appreciate less traffic of any kind, and so do networks. Without routers, packet traffic could choke the entire network. Most traffic will stay on the local network—only packets destined for other networks will pass through the router and over to another subnet. This traffic reduction also improves overall performance.

Simplified management It's easier to identify and isolate network problems in a group of smaller networks connected together than within one gigantic one.

The original designers of the IP protocol envisioned a small Internet with only mere tens of networks and hundreds of hosts. Their addressing scheme used a network address for each physical network. As you can imagine, this scheme and the unforeseen growth of the Internet created a few problems.

To name one, a single network address can be used to refer to multiple physical networks. An organization can request individual network addresses for each one of its physical networks. If these requests were granted, there wouldn't be enough addresses to go around. Another problem relates to routers—if each router on the Internet needed to know about every physical network, routing tables would be impossibly huge. There would be an overwhelming amount of administrative overhead to maintain those tables, and the resulting physical overhead on the routers would be massive (CPU cycles, memory, disk space, and so on). Because routers exchange routing information with each other, an additional, related consequence is that a terrific overabundance of network traffic would result.

Although there's more than one way to approach this problem, the principal solution is the one that will be covered in this book—subnetting. As you might guess, subnetting is the process of carving a single IP network into smaller logical subnetworks. This trick is achieved by subdividing the host portion of an IP address to create something called a *subnet address*. The actual subdivision is accomplished through the use of a *subnet mask*—more on that later.

In the following sections, you will see exactly how to calculate and apply subnetting.

Implementing Subnetting

Before you can implement subnetting, you need to determine your current requirements and plan on how best to implement your subnet scheme. Follow these guidelines:

- Determine the number of required network IDs: one for each subnet and one for each WAN connection.
- Determine the number of required host IDs per subnet: one for each TCP/IP device, including computers, network printers, and router interfaces.

Based on these two data points, create the following:

- One subnet mask for your entire network
- A unique subnet ID for each physical segment
- A range of host IDs for each unique subnet

An organization with a single network address can have a subnet address for each individual physical network. It's important to remember that each subnet is still part of the shared network address but it also has an additional identifier denoting its individual subnetwork number. This identifier is called a subnet address. For example, consider a hotel or office building. Say a hotel has 1000 rooms, with 75 rooms to a floor. You start at the first room on the first floor and number it 1. When you get to the first room on the second floor, you number it 76, and you would keep going until you reach room 1000. Now someone looking for room 521 would have to guess approximately which floor the room was on. If you were to “subnet” the hotel, you would identify the first room on the first floor with the number 101 (1 = Floor 1 and 01 = Room 1), the first room on the second floor with 201, and so on. The guest looking for room 521 would go to the 5th floor and look for room 21.

Subnetting solves several addressing problems. First, if an organization has several physical networks but only one IP network address, it can handle the situation by creating subnets. Next, because subnetting allows many physical networks to be grouped together, fewer entries in a routing table are required, notably reducing network overhead. Finally, these things combine to collectively yield greatly enhanced network efficiency.

Next, you will see how you can benefit from the features of subnetting in your network.

How to Hide Information

One benefit of subnetting is the ability to hide your address scheme from the outside world. Suppose that the Internet refers to Widget, Inc. only by its single network address, 130.57. Suppose as well that Widget, Inc. has several divisions and each is an independent business unit. Because Widget's network administrators have implemented subnetting, the Widget routers use the

subnet addresses to route the packets to the correct internal subnet when packets come into its network. Thus, the complexity of Widget, Inc.'s network can be hidden from the rest of the Internet. This is called *information hiding*. Routers on the Internet see only one external address for the Widget network.

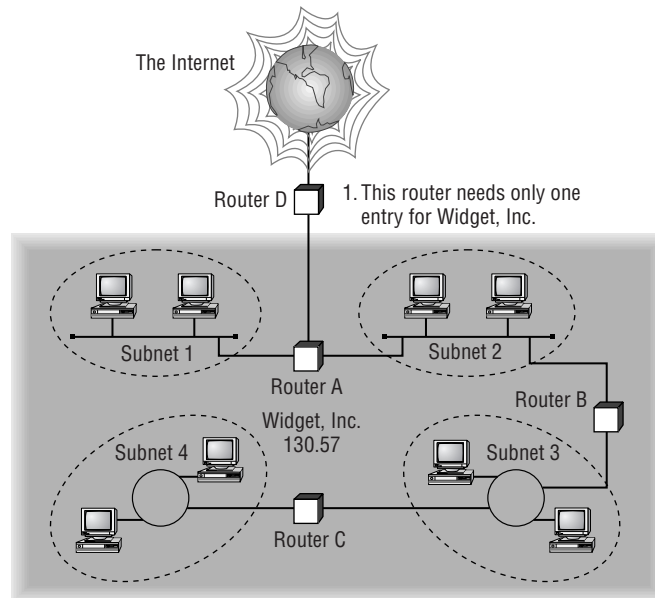
Information hiding also benefits the routers inside the Widget network. Without subnets, each Widget router would need to know the address of each machine on the entire Widget network—causing additional overhead and poor routing performance. The subnet scheme eliminates the need for each router to know about every machine on the entire Widget network; their routers need only the following two types of information:

- The addresses of the subnets to which they are attached
- The other subnet addresses

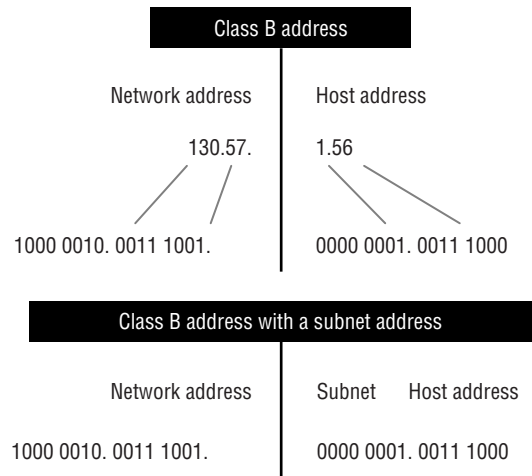
How to Implement Subnetting

Subnetting is implemented by assigning a subnet address to each machine on a given physical network. For example, in Figure 1.14, each machine on Subnet 1 has a subnet address of 1.

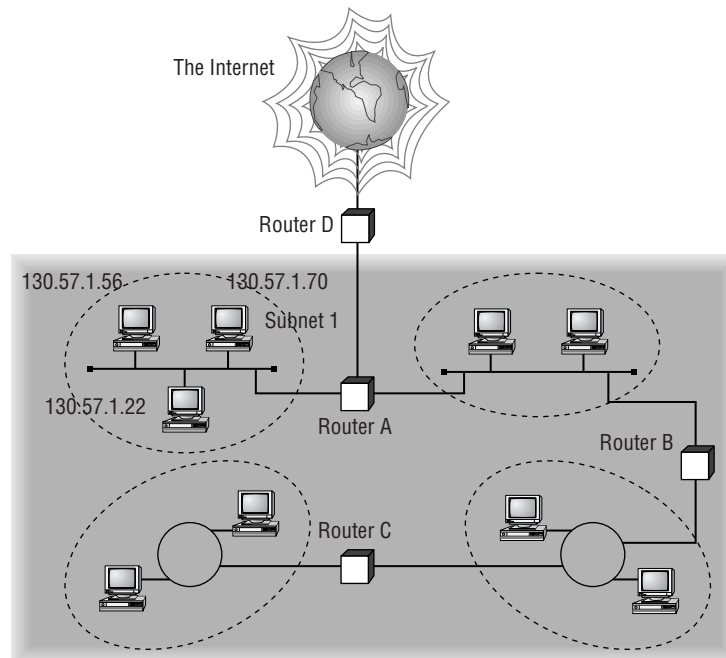
FIGURE 1.14 A sample subnet



The network portion of an IP address can't be altered. Every machine on a particular network must share the same network address. In Figure 1.14, you can see that all of Widget, Inc.'s machines have a network address of 130.57. That principle is constant. In subnetting, it's the host address that's manipulated; the network address doesn't change. The subnet address scheme takes a part of the host address and recycles it as a subnet address. Bit positions are stolen from the host address to be used for the subnet identifier. Figure 1.15 shows how an IP address can be given a subnet address.

FIGURE 1.15 Network vs. host addresses

Because the Widget, Inc. network is a Class B, the first 2 bytes specify the network address and are shared by all machines on the network—regardless of their particular subnet. Here, every machine's address on the subnet must have its third byte read 0000 0001. The fourth byte, the host address, is the unique number that identifies the actual host within that subnet. Figure 1.16 illustrates how a network address and a subnet address can be used together.

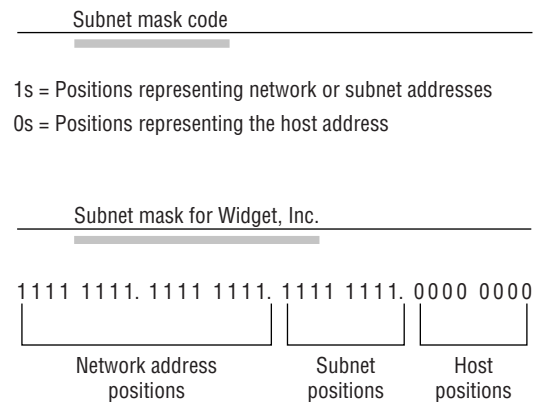
FIGURE 1.16 The network address and its subnet

How to Use Subnet Masks

For the subnet address scheme to work, every machine on the network must know which part of the host address will be used as the subnet address. This is accomplished by assigning each machine a subnet mask.

The network administrator creates a 32-bit subnet mask comprising 1s and 0s. The 1s in the subnet mask represent the positions that refer to the network or subnet addresses. The 0s represent the positions that refer to the host part of the address. This combination is illustrated in Figure 1.17.

FIGURE 1.17 The subnet mask revealed



In our Widget, Inc. example, the first two bytes of the subnet mask are 1s because Widget's network address is a Class B address, formatted as **Network.Network.Node.Node**. The third byte, normally assigned as part of the host address, is now used to represent the subnet address. Hence, those bit positions are represented with 1s in the subnet mask. The fourth byte is the only part in our example that represents the unique host address.

The subnet mask can also be expressed using the decimal equivalents of the binary patterns. The binary pattern of 1111 1111 is the same as decimal 255. Consequently, the subnet mask in our example can be denoted in two ways, as shown in Figure 1.18.

FIGURE 1.18 Different ways to represent the same mask

Subnet mask in binary: 1111 1111. 1111 1111. 1111 1111. 0000 0000

Subnet mask in decimal: 255 . 255 . 255 . 0

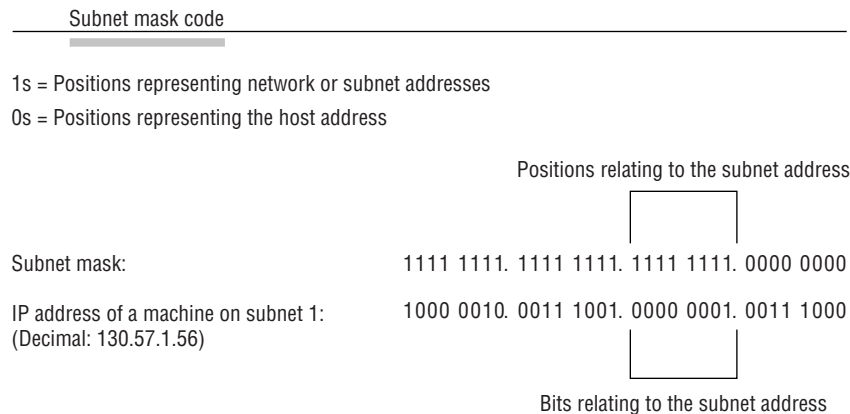
(The spaces in the above example are only for illustrative purposes.
The subnet mask in decimal would actually appear as 255.255.255.0.)

Not all networks need to have subnets and therefore don't need to use subnet masks. In this case, they are said to have a *default subnet mask*. This is basically the same as saying they don't have a subnet address. The default subnet masks for the different classes of networks are shown in Table 1.4.

TABLE 1.4 Special Network Addresses

Class	Format	Default Subnet Mask
A	Network.Node.Node.Node	255.0.0.0
B	Network.Network.Node.Node	255.255.0.0
C	Network.Network.Network.Node	255.255.255.0

Once the network administrator has created the subnet mask and assigned it to each machine, the IP software applies the subnet mask to the IP address to determine its subnet address. The word *mask* carries the implied meaning of “lens” in this case—the IP software looks at its IP address through the lens of its subnet mask to see its subnet address. An illustration of an IP address being viewed through a subnet mask is shown in Figure 1.19.

FIGURE 1.19 Applying the subnet mask

In this example, the IP software learns through the subnet mask that, instead of being part of the host address, the third byte of its IP address is now going to be used as a subnet address. The IP software then looks in its IP address at the bit positions that correspond to the mask, which are 0000 0001.

The final step is for the subnet bit values to be matched up with the binary numbering convention and converted to decimal. In the Widget, Inc. example, the binary-to-decimal conversion is simple, as illustrated in Figure 1.20.

By using the entire third byte of a Class B address as the subnet address, it is easy to set and determine the subnet address. For example, if Widget, Inc. wants to have a Subnet 6, the third byte of all machines on that subnet will be 0000 0110 (decimal 6 in binary).

Using the entire third byte of a Class B network address for the subnet allows for a fair number of available subnet addresses. One byte dedicated to the subnet provides eight bit positions.

Each position can be either a 1 or a 0, so the calculation is 2^8 , or 256. Because you cannot use the two patterns of all 0s and all 1s, you must subtract 2 for a total of 254. Thus, Widget, Inc. can have up to 254 total subnetworks, each with up to 254 hosts.

FIGURE 1.20 Converting the subnet mask to decimal

Binary numbering convention							
Position/value:	← (continued)	128	64	32	16	8	4 2 1
Widget third byte:		0	0	0	0	0	0 0 1
Decimal equivalent:		0 + 1 = 1					
Subnet address:		1					

Although the official IP specification limits the use of 0 as a subnet address, some products actually permit this usage. Microsoft’s TCP/IP stack allows it, as does the software in most routers (provided you enable this feature). This gives you one additional subnet. However, you should not use a subnet of 0 (all 0s) unless all of the software on your network recognizes this convention.

How to Calculate the Number of Subnets

The formulas for calculating the maximum number of subnets and the maximum number of hosts per subnet are as follows:

- $2^{\text{number of masked bits in subnet mask}} - 2 = \text{maximum number of subnets}$
- $2^{\text{number of unmasked bits in subnet mask}} - 2 = \text{maximum number of hosts per subnet}$

In the formulas, *masked* refers to bit positions of 1, and *unmasked* refers to positions of 0. The downside to using an entire byte of a node address as your subnet address is that you reduce the possible number of node addresses on each subnet. As explained earlier, without a subnet, a Class B address has 65,534 unique combinations of 1s and 0s that can be used for node addresses.

If you use an entire byte of the node address for a subnet, you then have only 1 byte for the host addresses, leaving only 254 possible host addresses. If any of your subnets will be populated with more than 254 machines, you have a problem. To solve it, you would then need to shorten the subnet mask, thereby lengthening the host address; this gives you more available host addresses on each subnet. A side effect of this solution is that it shrinks the number of possible subnets.

Figure 1.21 shows an example of using a smaller subnet address. A company called Acme, Inc. expects to need a maximum of 14 subnets. In this case, Acme does not need to take an entire byte from the host address for the subnet address. To get its 14 different subnet addresses, it needs to snatch only 4 bits from the host address ($2^4 - 2 = 14$). The host portion of the address has 12 usable bits remaining ($2^{12} - 2 = 4094$). Each of Acme’s 14 subnets could then potentially have a total of 4094 host addresses; 4094 machines on each subnet should be plenty.

FIGURE 1.21 An example of a smaller subnet address

<u>Acme, Inc.</u>	
Network address:	132.8 (Class B; net.net.host.host)
Example IP address:	1000 0100. 0000 1000. 0001 0010. 0011 1100
Decimal:	132 . 8 . 18 . 60
<u>Subnet Mask Code</u>	
1s = Positions representing network or subnet addresses	
0s = Positions representing the host address	
Subnet mask:	
Binary:	1111 1111. 1111 1111. 1111 0000. 0000 0000
Decimal:	255 . 255 . 240 . 0
(The decimal 240 is equal to the binary 1111 0000.)	
Positions relating to the subnet address	
Subnet mask:	1111 1111. 1111 1111. 1111 0000. 0000 0000
IP address of a Acme machine:	1000 0100. 0000 1000. 0001 0010. 0011 1100
(Decimal: 132.8.18.60)	
Bits relating to the subnet address	
<u>Binary-to-Decimal Conversions for Subnet Address</u>	
Subnet mask positions:	1 1 1 1 0 0 0 0
	↓ ↓ ↓ ↓
Position/value: ← (continue)	128 64 32 16 8 4 2 1
Third byte of IP address:	0 0 0 1 0 0 1 0
Decimal equivalent:	0 + 16 = 16
Subnet address for this IP address:	16

Applying Subnetting

Sometimes subnetting can be confusing. It can be quite difficult to remember all those numbers. You can step back a minute and take a look at the primary classes of networks and how to subnet each one. You'd start with Class C because it uses only 8 bits for the node address, so it's the easiest to calculate. In the following sections, we will look at how to subnet the various types of networks.

Class C

If you recall, a Class C network uses the first 3 bytes (24 bits) to define the network address. This leaves you 1 byte (8 bits) with which to address hosts. So, if you want to create subnets, your options are limited because of the small number of bits left available.

If you break down your subnets into chunks smaller than the default Class C, then figuring out the subnet mask, network number, broadcast address, and router address can be kind of confusing. Table 1.5 summarizes how you can break a Class C network down into one, two, four, or eight smaller subnets, with the subnet masks, network numbers, broadcast addresses, and router addresses. The first 0 bytes have simply been designated x.y.z. (Note that the table assumes you can use the all-0 subnet, too.)

TABLE 1.5 Class C Subnets

Number of Desired Subnets	Subnet Mask	Network Number	Router Address	Broadcast Address	Remaining Number of IP Addresses
1	255.255.255.0	x.y.z.0	x.y.z.1	x.y.z.255	253
2	255.255.255.128	x.y.z.0	x.y.z.1	x.y.z.127	125
	255.255.255.128	x.y.z.128	x.y.z.129	x.y.z.255	125
4	255.255.255.192	x.y.z.0	x.y.z.1	x.y.z.63	61
	255.255.255.192	x.y.z.64	x.y.z.65	x.y.z.127	61
	255.255.255.192	x.y.z.128	x.y.z.129	x.y.z.191	61
	255.255.255.192	x.y.z.192	x.y.z.193	x.y.z.255	61
8	255.255.255.224	x.y.z.0	x.y.z.1	x.y.z.31	29
	255.255.255.224	x.y.z.32	x.y.z.33	x.y.z.63	29
	255.255.255.224	x.y.z.64	x.y.z.65	x.y.z.95	29
	255.255.255.224	x.y.z.96	x.y.z.97	x.y.z.127	29
	255.255.255.224	x.y.z.128	x.y.z.129	x.y.z.159	29
	255.255.255.224	x.y.z.160	x.y.z.161	x.y.z.191	29
	255.255.255.224	x.y.z.192	x.y.z.193	x.y.z.223	29
	255.255.255.224	x.y.z.224	x.y.z.225	x.y.z.255	29

For example, suppose you want to chop up a Class C network, 200.211.192.x, into two subnets. As you can see in the table, you'd use a subnet mask of 255.255.255.128 for each subnet. The first subnet would have network number 200.211.192.0, router address 200.211.192.1, and broadcast address 200.211.192.127. You could assign IP addresses 200.211.192.2 through 200.211.192.126—that's 125 different IP addresses. (Notice that heavily subnetting a network results in the loss of a progressively greater percentage of addresses to the network number, broadcast address, and router address.) The second subnet would have network number 200.211.192.128, router address 200.211.192.129, and broadcast address 200.211.192.255.

Now, you may be wondering how you can subnet a Class C network as in Table 1.5. If you use the $2^x - 2$ calculation, the subnet 128 in the table doesn't make sense. It turns out that there's a legitimate reason to do it this way:

1. Remember that using subnet zero is not allowed according to the RFCs, but by using it you can subnet your Class C network with a subnet mask of 128. This uses only 1 bit, and according to your calculator, $2^1 - 2 = 0$, giving you zero subnets.
2. By using a router that supports subnet zero, you can assign 1–127 for hosts and 129–254, for hosts, as stated in the table. This saves a bunch of addresses! If you were to stick to the method defined by RFC standards, the best you could gain is a subnet mask of 192 (2 bits), which allows you only two subnets ($2^2 - 2 = 2$).

To determine the first subnet number, subtract the subnet mask from 256. Our example yields the following equation: $256 - 192 = 64$. So 64 is your first subnet.

To determine a second subnet number, add the first subnet number to itself. To determine a third subnet number, add the first subnet number to the second subnet number. To determine a fourth subnet number, add the first subnet number to the third subnet number. Keep adding the first subnet number in this fashion until you reach the actual subnet number. For example, 64 plus 64 equals 128, so your second subnet is 128. And 128 plus 64 is 192. Because 192 is the subnet mask, you cannot use it as an actual subnet. This means your valid subnets are 64 and 128.

The numbers between the subnets are your valid hosts. For example, the following are valid hosts in a Class C network with a subnet mask of 192:

- The valid hosts for subnet 64 are in the range 65–126, which gives you 62 hosts per subnet (using 127 as a host would mean your host bits would be all 1s). That's not allowed because the all-1s format is reserved as the broadcast address for that subnet.
- The valid hosts for subnet 128 are in the range 129–190. you might be asking yourself what happened to 191–254? The subnet mask is 192, which you cannot use, and 191 would be all 1s and used as the broadcast address for this subnet. Anything above 192 is also invalid for this subnet because these are automatically lost through the subnetting process.

As you can see, this solution wastes a lot of addresses: 130 to be exact. In a Class C network, this would certainly be hard to justify—the 128 subnet is a much better solution if you only need two subnets.

But what happens if you need four subnets in your Class C network?

By using the calculation of $2^x \text{ number of masked bits} - 2$, you would need 3 bits to get six subnets ($2^3 - 2 = 6$). What are the valid subnets and what are the valid hosts of each subnet? Let's figure it out.

11100000 is 224 in binary and would be the subnet mask. This must be the same on all workstations.



You're likely to see test questions that ask you to identify the problem with a given configuration. If a workstation has the wrong subnet mask, the router could "think" the workstation is on a different subnet than it actually is. When that happens, the misguided router won't forward packets to the workstation in question. Similarly, if the mask is incorrectly specified in the workstation's configuration, that workstation will observe the mask and send packets to the default gateway when it shouldn't.

To figure out the valid subnets, subtract the subnet mask from 256; $256 - 224 = 32$, so 32 is your first subnet. The other subnets would be 64, 96, 128, 160, and 192. The valid hosts are the numbers between the subnet numbers, except the numbers that equal all 1s. These numbers would be 63, 95, 127, 159, 191, and 223. Remember that using all 1s is reserved for the broadcast address of each subnet.

The valid subnets and hosts are as follows:

Subnet	Hosts
32	33–62
64	65–94
96	97–126
128	129–158
160	161–190
192	193–222

You can add one more bit to the subnet mask just for fun. You were using 3 bits, which gave you 224. By adding the next bit, the mask now becomes 240 (11110000).

By using 4 bits for the subnet mask, you get 14 subnets because $2^4 - 2 = 14$. This subnet mask also gives you only 4 bits for the host addresses, or 14 hosts per subnet. As you can see, the amount of hosts per subnet gets reduced rather quickly when subnetting a Class C network.

The first valid subnet for subnet 240 is 16 ($256 - 240 = 16$). Your subnets are then 16, 32, 48, 64, 80, 96, 112, 128, 144, 160, 176, 192, 208, and 224. Remember that you cannot use the actual subnet number as a valid subnet, so 240 is invalid as a subnet number. The valid hosts are the numbers between the subnets, except for the numbers that are all 1s—the broadcast address for the subnet.

The following are valid subnets and hosts:

Subnet	Hosts
16	17–30
32	33–46

Subnet	Hosts
48	49–62
64	65–78
80	81–94
96	97–110
112	113–126
128	129–142
144	145–158
160	161–174
176	177–190
192	193–206
208	209–222
224	225–238

Class B

Because a Class B network has 16 bits for host addresses, you have plenty of available bits to play with when figuring out a subnet mask. Remember that you have to start with the leftmost bit and work toward the right. For example, a Class B network would look like X.Y.0.0, with the default mask of 255.255.0.0. Using the default mask would give you one network with 65,536 hosts.

The default mask in binary is 11111111.11111111.00000000.00000000. The 1s represent the network, and the 0s represent the hosts. So when creating a subnet mask, the leftmost bit(s) will be borrowed from the host bits (0s, not 1s) to become the subnet mask. You use the remaining available bits for hosts.

If you use only 1 bit, you have a mask of 255.255.128.0. This mask will be somewhat harder to subnet than the Class C 128 subnet mask. With 16 bits, you typically don't need to worry about a shortage of host IDs, so using 128 just isn't worth the trouble. The first mask you should use is 255.255.192.0, or 11111111.11111111.11000000.00000000.

You now have three parts of the IP address: the network address, the subnet address, and the host address. A 192 mask is figured out the same way as a Class C network address, but this time you'll end up with a lot more hosts.

There are two subnets because $2^2 - 2 = 2$. The valid subnets are 64 and 128 ($256 - 192 = 64$ and $64 + 64 = 128$). However, there are 14 bits (0s) left over for host addressing. This gives you 16,382 hosts per subnet ($2^{14} - 2 = 16,382$).

The valid subnets and hosts are as follows:

Subnet	Hosts
64	X.Y.64.1 through X.Y.127.254
128	X.Y.128.1 through X.Y.191.254

You can add another bit to the subnet mask, making it 11111111.11111111.11100000.00000000 or 255.255.224.0. There are six subnets ($2^3 - 2 = 6$). The valid subnets are 32, 64, 96, 128, 160, and 192 ($256 - 224 = 32$). The valid hosts are listed here:

Subnet	Hosts
32	X.Y.32.1 through 63.254
64	X.Y.64.1 through 95.254
96	X.Y.96.1 through 127.254
128	X.Y.128.1 through 159.254
160	X.Y.160.1 through 191.254
192	X.Y.192.1 through 223.254

Therefore, if you use a 255.255.224.0 subnet mask, you can create six subnets, each with 8190 hosts.

You can add a few more bits to the subnet mask and see what happens. If you use 9 bits for the mask, it gives you 510 subnets ($2^9 - 2 = 510$). With only 7 bits for hosts, you still have 126 hosts per subnet ($2^7 - 2 = 126$). The mask looks like this:

11111111.11111111.11111111.10000000 or 255.255.255.128

You could add even more bits and see what you get. If you use 14 bits for the subnet mask, you get 16,382 subnets ($2^{14} - 2 = 16382$), but this gives you only two hosts per subnet ($2^2 - 2 = 2$). The subnet mask would look like this:

11111111.11111111.11111111.11111100 or 255.255.255.252

You may be wondering why you would ever use a 14-bit subnet mask with a Class B address. This approach is actually very common. Think about having a Class B network and using a subnet mask of 255.255.255.0. You'd have 254 subnets and 254 hosts per subnet. Imagine also that you have a network with many WAN links. Typically, you'd have a direct connection between each site. Each of these links must be on its own subnet or network. There will be two hosts on these subnets—one address for each router port. If you used the mask described earlier (255.255.255.0), you would waste 252 host addresses per subnet. Using the 255.255.255.252 subnet mask, you have many, many subnets available—each with only two hosts.

You can use this approach only if you are running a routing algorithm like Enhanced Interior Gateway Routing Protocol (EIGRP) or Open Shortest Path First (OSPF), which we will talk about later in this book. These routing protocols allow what is called Variable Length Subnet Masks (VLSMs). VLSM allows you to run the 255.255.255.252 subnet mask on your interfaces to the WANs and run 255.255.255.0 on your router interfaces in your LAN. It works because routing protocols like EIGRP and OSPF transmit the subnet mask information in the update packets that it sends to the other routers. RIP doesn't transmit the subnet mask and therefore cannot use VLSM.

Class A

Class A networks have a ton of bits available. A default Class A network subnet mask is only 8 bits, or 255.0.0.0, giving you a whopping 24 bits for hosts to play with.

If you use a mask of 11111111.11111111.00000000.00000000, or 255.255.0.0, you'll have 8 bits for subnets, or 254 subnets ($2^8 - 2 = 254$). This leaves 16 bits for hosts, or 65,534 hosts per subnet ($2^{16} - 2 = 65534$). Instead, you could split the 24 bits evenly between subnets and hosts, giving each one 12 bits. The mask would look like this: 11111111.11111111.11110000.00000000, or 255.255.240.0. How many valid subnets and hosts would you have?

The answer is 4094 subnets each with 4094 hosts ($2^{12} - 2 = 4094$). Knowing which hosts and subnets are valid is a lot more complicated than it was for either Class B or C networks.

The second octet will be somewhere between 1 and 254. However, the third octet you will need to figure out. Because the third octet has a 240 mask, you'll get 16 ($256 - 240 = 16$) as your base subnet number. The third octet must start with 16 and will be the first subnet, the second subnet will be 32, and so on. This means that your valid subnets are

X.1-254.16.1 through X.1-254.31.254

X.1-254.32.1 through X.1-254.47.254

X.1-254.48.1 through X.1-254.63.254

and so on for the remaining bits.



New to Microsoft is the way that address ranges are written. For example, an address of 131.107.2.0 with a subnet mask of 255.255.255.0 is listed as 131.107.2.0/24 because the subnet mask contains 24 1s. An address listed as 141.10.32.0/19 would have a subnet mask of 255.255.224.0, or 19 1s (default subnet mask for a Class B plus 3 bits). This is the new nomenclature used in all Microsoft exams and is referred to as Classless Inter-Domain Routing (CIDR) notation.

Summary

The following list includes some of the important topics covered in this chapter:

- How the OSI networking model is organized into seven layers: Physical, Data-Link, Network, Transport, Session, Presentation, and Application.
- What each level of the OSI stack does. The Physical layer is responsible for sending bits from one computer to another. The Data-Link layer provides for the flow of data over a single physical link from one device to another. The Network layer moves packets between devices that are more than one link away from each other. The Transport layer ensures that data is delivered error free, in sequence, and with no losses or duplications. The Session layer allows applications on separate computers to share a connection called a session. The Presentation layer translates data between the formats the network requires and the formats the computer expects. Finally, the Application layer is the topmost layer of the OSI model, and it provides services that directly support user applications, such as database access, e-mail, and file transfers.
- Windows Server 2003 includes support for AppleTalk, NWLink, and TCP/IP. TCP/IP is the primary protocol in use today, and Microsoft encourages you to use TCP/IP exclusively, if possible.

- The 32-bit IP address is a structured or hierarchical address that is used to uniquely identify every machine on a network. You learned how to determine available IP addresses and implement subnetting.

Exam Essentials

Understand what subnetting is and when to use it. If an organization is large and has many computers, or if its computers are geographically dispersed, it's sensible to divide its large network into smaller ones connected by routers. These smaller networks are called subnets. Subnetting is the process of carving a single IP network into smaller, logical subnetworks.

Understand subnet masks. For the subnet address scheme to work, every machine on the network must know which part of the host address will be used as the subnet address. The network administrator creates a 32-bit subnet mask consisting of 1s and 0s. The 1s in the subnet mask represent the positions that refer to the network or subnet addresses. The 0s represent the positions that refer to the host portion of the address.

Key Terms

Before you take the exam, be certain you are familiar with the following terms:

binding	Open Systems Interconnection (OSI)
cyclic redundancy check (CRC)	OSI model
default subnet mask	packet payload
hierarchical address	packets
information hiding	protocol stack
Internet Protocol (IP)	Service Access Points (SAPs)
Logical Link Control (LLC)	subnet address
Media Access Control (MAC)	subnet mask
network address	subnets
Network Driver Interface Specification (NDIS)	TCP/IP
node address	Transmission Control Protocol (TCP)
NWLink	User Datagram Protocol (UDP)
Open Data-link Interface (ODI)	

Review Questions

1. You have a large IP routed network using the address 137.25.0.0; it is composed of 20 subnets, with a maximum of 300 hosts on each subnet. Your company continues on a merger and acquisitions spree, and your manager has told you to prepare for an increase to 50 subnets, with some of them containing more than 600 hosts. Using the existing network address, which of the following subnet masks would work for that requirement from your manager?
 - A. 255.255.252.0
 - B. 255.255.254.0
 - C. 255.255.248.0
 - D. 255.255.240.0
2. You are brought into a small company that occupies two floors of a building that has had two separate networks for some time. The company now wants these two networks to share some information files such as documents, spreadsheets, and databases. You determine that one of the networks is a NetWare 3.x LAN and the other is a Windows NT peer-to-peer network running NetBEUI. The NetWare LAN has a printer that you want users on the NT network to be able to use. Other than any client software, what protocol will you have to install for the Windows NT workstations to be able to access the NetWare printer?
 - A. NetBEUI
 - B. TCP/IP
 - C. AppleTalk
 - D. NWLink
 - E. DLC
3. The company you work for is growing dramatically via acquisitions of other companies. As the network administrator, you need to keep up with the changes because they affect the workstations and you need to support them. When you started, there were 15 locations connected via routers, and now there are 25. As new companies are acquired, they are migrated to Windows Server 2003 and brought into the same domain as another site. Management says that they are going to acquire at least 10 more companies in the next 2 years. The engineers have also told you that they are redesigning the company's Class B address into an IP addressing scheme that will support these requirements and that there will never be over 1000 network devices on any subnet. What will be the appropriate subnet mask to support this network when the changes are completed?
 - A. 255.255.252.0
 - B. 255.255.248.0
 - C. 255.255.255.0
 - D. 255.255.255.128

4. You work for a small printing company that has 75 workstations. Most of them run standard office applications like word processing, spreadsheet, and accounting programs. Fifteen of the workstations are constantly processing huge graphics files and then sending print jobs to industrial-size laser printers. The performance of the network has always been an issue, but you have never addressed it. You have now migrated your network to Windows XP and Windows Server 2003 and have decided to take advantage of the routing capability built into the Windows Server 2003. You choose the appropriate server and place two NICs in the machine, but you realize that you have only one network address, 201.102.34.0, which you obtained years ago. How should you subnet this address to segment the bandwidth hogs from the rest of the network while giving everyone access to the entire network?
- A. 255.255.255.192
 - B. 255.255.255.224
 - C. 255.255.255.252
 - D. 255.255.255.240
5. A packet is sent from one computer to another across a network. Various protocols move the packet down the OSI stack from the sending computer and up the OSI stack to the receiving computer. How do the protocols know where to send the packet?
- A. Each packet has a trailer that contains source and destination addresses.
 - B. Each packet has a header that contains an alert signal and source and destination addresses.
 - C. The data portion of every packet stores all the source and destination information.
 - D. Special packets, called header packets, that contain only source and destination addresses are sent first. Every packet that follows the header packet is sent to the destination address contained in the header packet.
6. You have been engaged at a large automobile manufacturing organization to help company officials understand and alleviate their network traffic overutilization rates. After discussions with the network administrator, you discover that the organization was initially running only TCP/IP but they recently installed NetBEUI as well. After some investigation, you are told the reason for the dual protocols: The administrator was told that they were going to have to support several NetBIOS applications on the network. To permit this NetBIOS support, the administrator then directed several staff members to add the NetBEUI stack to the workstations and, in the bindings, to place NetBEUI first. Now the network is having performance problems. What mistake did this administrator make?
- A. Because NetBEUI was placed at the top of the binding list, the workstations are trying to communicate through NetBEUI first, even though the resource to which they are trying to connect uses only TCP/IP.
 - B. Because NetBEUI was placed at the top of the binding list, NetBEUI is being used to communicate with the TCP/IP-based servers and is less efficient in communicating with a different protocol.
 - C. NetBEUI is unnecessary for a client's communication with a NetBIOS program.
 - D. Although TCP/IP needs NetBEUI in order to communicate with a NetBIOS program, NetBEUI needs to be bound directly to the TCP/IP stack so that they will work together properly.

7. The Integrated Network Computing company is a software development house that writes small utility programs for a wide range of networks. In addition to supporting their Windows Server 2003 network, you are responsible for verifying that some of the applications that are developed function properly. During these tests, you have to install transport protocols from other development houses that are used by various systems. You have worked out the issues surrounding the different protocols working on Windows Server 2003 by requiring the protocol developers to make sure their protocols are compliant with what standard?
- A. ODI
 - B. DLC
 - C. NDIS
 - D. NetBIOS
8. You work for Carpathian Worldwide Enterprises, which has more than 50 administrative and manufacturing locations around the world. The size of these organizations varies greatly, with the number of computers per location ranging from 15 to slightly fewer than 1000. The sales operations use more than 1000 facilities, each of which contains 2 to 5 computers. Carpathian is also in merger talks with another large organization; if the merger materializes as planned, you will have to accommodate another 100 manufacturing and administrative locations, each with a maximum of 600 computers, as well as 2,000 additional sales facilities. You don't have any numbers for the future growth of the company, but you are told to keep growth in mind. You decide to implement a private addressing plan for the entire organization. More than half of your routers don't support Variable Length Subnet Masking. What subnet masks would work for this situation? (Choose all that apply.)
- A. 255.255.224.0
 - B. 255.255.240.0
 - C. 255.255.248.0
 - D. 255.255.252.0
 - E. 255.255.254.0
9. You administer a very large network that consists of Windows 2000 and XP Professional and Windows Server 2003 computers. You want to implement DNS, DHCP, and WINS, and every computer must have access to the Internet and services on non-Windows machines. You want to be able to configure the network from a central location. Which network protocol provides the ability to do all these things?
- A. NetBEUI
 - B. NWLink
 - C. TCP
 - D. TCP/IP

- 10.** You are the administrator for a Windows NT network that has been internally focused on basic file and print services. You have been charged with upgrading your network to Windows Server 2003 and also allowing the users of the network to find information on the Internet. Currently, the network is running NWLink because of routing needs between two locations and a lack of IP experience. You need to change the network protocol to TCP/IP to support Internet connectivity. What primary layers in the OSI model do you need to consider to allow the workstations to access the Internet for simple browsing? (Choose all that apply.)
- A.** Physical layer
 - B.** Network layer
 - C.** Application layer
 - D.** Presentation layer
 - E.** Transport layer
- 11.** You have just been asked to troubleshoot intermittent communication problems on a fairly old network for a company that builds and repairs elevator motors. You have determined that the network is a straightforward thin-coax Ethernet Windows NT LAN running TCP/IP. The company wants to upgrade to Windows Server 2003, hoping that the now-stable platform will resolve the intermittent problems. You perform the upgrade; all goes smoothly, and initially everything seems to function properly. However, the intermittent problems show up again. What layer in the OSI model is the most likely place for the problems to be occurring?
- A.** Physical layer
 - B.** Data-Link layer
 - C.** Network layer
 - D.** Transport layer
 - E.** Session layer
- 12.** You are working at a manufacturing company that occupies an entire city block. Management informs you that they have acquired another business on the other side of town that previously had been a supplier to your company. The Windows Server 2003 network that you have been supporting now needs to be connected to the new location through a router. You also have several NetBIOS applications that need to continue functioning properly. What protocols are available for you to use to ensure that these criteria are met? (Choose all that apply.)
- A.** NWLink
 - B.** TCP/IP
 - C.** XNS
 - D.** NetBEUI




13. The company you work for manufactures handballs and has an Intel PC-based Windows Server 2003 network. To cut packaging costs, the management of the company has acquired a graphics arts company. Its network is entirely Macintosh based and is currently using AppleTalk as the protocol to communicate among workstations. You have to integrate the two networks so that they can easily share information. What protocols must you have on your network for communication among all the workstations on this network?

A. AppleTalk
B. TCP/IP
C. NWLink
D. NetBEUI

14. You administer a network that contains 175 machines. Your manager has assigned the network the IP address 192.168.11.0 with the default subnet mask of 255.255.255.0. A router that has one WAN interface and eight LAN interfaces connects this network to the corporate WAN.

You want to subnet the network into three subnets, and you want to reserve a few addresses for a fourth subnet, just in case you need it later. You decide that Subnet A will contain 25 computers, Subnet B will contain 50 computers, and Subnet C will contain 100 computers.

In the following exhibit, select the network addresses and subnet masks in the Choices column and place them in the appropriate boxes in the other three columns. Each item may be used only once.

Choices:	 Subnet A	 Subnet B	 Subnet C
192.168.11.128			
255.255.255.128			
192.168.11.0			
255.255.255.192			
192.168.11.192			
255.255.255.224			
Network address	<input type="text"/>	<input type="text"/>	<input type="text"/>
Subnet mask	<input type="text"/>	<input type="text"/>	<input type="text"/>

15. Your multinational company has a Windows NT and Novell NetWare network that is built on several subnetworks. To provide interoperability, you have been using NWLink on the NT network and IPX for the NetWare network. You have been told that the Windows NT network must be migrated to Windows Server 2003 because it's less expensive to administer. You know that the administrative cost benefits are a result of utilizing Active Directory, so you include this service in your migration plan. What are you going to have to do immediately in order to install and begin using Active Directory on this network?

A. Change the protocol to TCP/IP.
B. Make sure that you install a copy of Active Directory on the NetWare servers as well as on the Windows 2003 Server computers.
C. As you upgrade the Windows NT servers, make sure that you choose to upgrade some of them as domain controllers so that you can install Active Directory on them.
D. Install NetBEUI in order to provide connectivity for the NetBIOS components of Windows Server 2003.




Answers to Review Questions

1. A. A Class B address with a default subnet mask of 255.255.0.0 will support up to 65,534 hosts. To increase the number of networks that this network will support, you need to subnet the network by borrowing bits from the host portion of the address. The subnet mask 255.255.252.0 uses 6 bits from the hosts area and will support 62 subnetworks while leaving enough bits to support 1022 hosts per subnet.
 The subnet mask 255.255.248.0 uses 5 bits from the hosts and will support 30 subnetworks while leaving enough bits to support 2046 hosts per subnet. 255.255.252.0 is probably the better answer because it leaves quite a bit of room for further growth in the number of networks while still leaving room for more than 1000 hosts per subnet, which is a fairly large number of devices on one subnet. The subnet mask 255.255.254.0 uses 7 bits from the hosts area and will support more than 120 networks, but it will leave only enough bits to support 500 hosts per subnet. The subnet mask 255.255.240.0 uses 4 bits from the hosts and will support only 14 subnetworks, even though it will leave enough bits to support more than 4000 hosts per subnet.
2. D. Older NetWare networks are based on the IPX protocol. There must be a common protocol in order for two network devices to communicate. Because there is no server to run the gateway for NetWare services, each NT workstation must have NWLink loaded. Also, the NetWare client must be installed on the workstations so that they will be able to connect to the printer.
3. A. The network mask applied to an address determines which portion of that address reflects the number of hosts available to that network. The balance with subnetting is always between the number of hosts and individual subnetworks that can be uniquely represented within one encompassing address. The number of hosts and networks that are made available depends upon the number of bits that can be used to represent them. This scenario requires more than 35 networks and fewer than 1000 workstations on each network. If you convert the subnet masks as described in the chapter, you will see that the mask in choice A allows for more than 60 networks and more than 1000 hosts. All of the other choices are deficient in either the number of networks or hosts that they represent.
4. A. The subnet mask 255.255.255.192 borrows 2 bits from the hosts, which allows you to build two separate networks that you can route through the Windows server. This will allow you to have 62 hosts on each segment. You'll be cutting it close, but this will work.
 The subnet mask 255.255.255.224 borrows 3 bits from the hosts; this allows you to create 6 networks, which you don't need, and leaves only enough bits for 30 hosts. The subnet mask 255.255.255.252 borrows 6 bits from the hosts; this allows you to create more than 60 networks, which you don't need, and leaves only enough bits for 2 hosts. The subnet mask 255.255.255.240 borrows 4 bits from the hosts; this allows you to create 15 networks, which you don't need, and leaves only enough bits for 15 hosts.
5. B. Each packet typically consists of three parts: a header, data, and a trailer. The header includes the source and destination addresses.

6. C. NetBEUI and NetBIOS are separate entities. NetBEUI is a transport protocol that has a NetBIOS interface. However, each protocol that comes with Windows Server 2003 has a NetBIOS component and can be used to communicate with NetBIOS programs. Although the binding order of protocols can have a performance effect on communication across the network, it has nothing to do with the problem described here. The extra protocol is simply consuming unnecessary network bandwidth.
7. C. Network Driver Interface Specification (NDIS) provides a standard way for protocols to bind to the data link drivers in Windows Server 2003. This is what allows Windows Server 2003 to support so many protocols. As long as a developer supports NDIS, the protocol will load in Windows Server 2003. However, this will not make it interoperate with the Windows Server 2003 services. The applications will have to be written to the specific protocols.
8. B, C, D. When you add up the locations that currently need to be given a network address, the total is 3150, and the maximum number of hosts at any one of these locations is less than 1000. The subnet masks need to support those requirements. The subnet masks given in options B, C, and D will provide the address space to support the outlined requirements. The subnet mask 255.255.240.0 supports more than 4000 subnets and 4000 hosts. The subnet mask 255.255.248.0 supports more than 8000 subnets and more than 2000 hosts. The subnet mask 255.255.252.0 supports more than 16,000 subnets and more than 1000 hosts.
 Although each of these subnet masks will work, at the rate that this company is growing, 255.255.252.0 is probably the best mask to prepare for the future. It's unlikely that there will ever be more than 1000 hosts on any given network. In fact, that number would probably cause performance problems on that subnet. Therefore, it's better to have more subnets available to deploy as the company grows.
 The subnet mask 255.255.224.0 supports more than 2000 subnets—an insufficient number to cover the locations. The subnet mask 255.255.254.0 supports more than 32,000 subnets but only 500 hosts per subnet, which are not enough hosts to cover all the locations.
9. D. TCP/IP is the most widely used protocol for interconnecting computers, and it is the only protocol used on the Internet. It works well with very large networks.
10. A, B, C, D, E. TCP sits at the Transport layer, and IP at the Network layer, and both are necessary to route requests through the Internet. However, you also need a browser such as Netscape or Internet Explorer to provide the HTTP calls to actually connect to the various websites; the browser sits at the Application layer. But any end-to-end communication uses all the levels of the OSI model at some point because each layer communicates with the layer below and the layer above to form the complete chain.
11. A. The Physical layer is concerned with signaling, specifically through electrical, optical, or radio signals. The high voltage associated with large motors can easily cause an interruption in the signaling of coax cable. There have been many cases of people running network cable through elevator shafts in a building because of their ease of access, only to have the network malfunction every time someone summons the car. The other layers are associated with software and are beyond the reach of most electrical interference unless it affects the entire workstation.

- 12.** A, B. Both NWLink and TCP/IP are routable and both can function properly with NetBIOS applications because they are both Microsoft's versions and have the interface for proper communication. XNS is a routable protocol but is not provided with Windows Server 2003, and with the overwhelming popularity of TCP/IP, XNS is generally no longer used in networks. NetBEUI, although it supports the NetBIOS programs, is not routable.
- 13.** B. Although Macintosh computers can use AppleTalk to communicate with each other, these computers can also run TCP/IP, so AppleTalk won't be necessary when these two networks are merged. You could add AppleTalk to the servers in the network, and the two machine types could share files back and forth, but if you can reduce the number of protocols on any network, it's a best practice to do so.

14.

			
	Subnet A	Subnet B	Subnet C
Network address	192.168.11.192	192.168.11.128	192.168.11.0
Subnet mask	255.255.255.224	255.255.255.192	255.255.255.128

The network address 192.168.11.192 with a subnet mask of 255.255.255.224 is perfect for Subnet A because it supports up to 30 hosts. The network address 192.168.11.128 with a subnet mask of 255.255.255.192 is perfect for Subnet B because it supports up to 62 hosts. The network address 192.168.11.0 with a subnet mask of 255.255.255.128 is perfect for Subnet C because it supports up to 126 hosts. That still leaves the network address 192.168.11.224 with a subnet mask of 255.255.255.224 available for a fourth subnet later.

- 15.** A. Active Directory requires TCP/IP in order to function. Even though you can have TCP/IP and IPX coexisting on the same network, it's not beneficial to have multiple protocols because they increase the level of support necessary for the network. Active Directory does not run on NetWare, and NetBEUI is not required for NetBIOS communication. Finally, Active Directory can be installed and uninstalled on any Windows Server 2003 computer. It's a service that is added rather than a particular type of server that is installed.

