Chapter 2

The Open Systems Interconnection Specifications

THE FOLLOWING COMPTIA NETWORK+ EXAM OBJECTIVES ARE COVERED IN THIS CHAPTER:

✓ **4.1 Explain the function of each layer of the OSI model**

- Layer 1 – physical
- Layer 2 – data link
- Layer 3 – network
- Layer 4 – transport
- Layer 5 – session
- Layer 6 – presentation
- Layer 7 – application

In this chapter, I'm going to dissect the Open Systems Interconnection (OSI) model and describe each part to you in detail, because you need a solid foundation on which to build your networking knowledge. The OSI model has seven hierarchical layers that were developed to enable different networks to communicate reliably between disparate systems.

Because this book is centering upon all things Network+, it's crucial for you to understand the OSI model as CompTIA sees it, so I'll present each of its seven layers in that light.

I'll finish this chapter with an introduction to encapsulation. *Encapsulation* is the process of encoding data as it goes down the OSI stack.

To find up-to-the-minute updates for this chapter, please see `www.lammle.com` or `www.sybex.com/go/comptianetwork+studyguide`.

Internetworking Models

When networks first came into being, computers could usually communicate only with computers from the same manufacturer. For example, companies ran either a complete DECnet solution or an IBM solution—not both together. In the late 1970s, the *Open Systems Interconnection (OSI) reference model* was created by the International Organization for Standardization (ISO) to break through this barrier.

The OSI model was meant to help vendors create interoperable network devices and software in the form of protocols so that different vendor networks could work with each other. Like world peace, it'll probably never happen completely, but it's still a great goal.

The OSI model is the primary architectural model for networks. It describes how data and network information are communicated from an application on one computer through the network media to an application on another computer. The OSI reference model breaks this approach into layers.

In the following section, I'll explain the layered approach and how you can use this approach to help troubleshoot internetworks.

The Layered Approach

Basically, a *reference model* is a conceptual blueprint of how communications should take place. It addresses all the processes required for effective communication and divides these processes into logical groupings called *layers*. When a communication system is designed in this manner, it's known as *layered architecture*.

Think of it like this: Say you and some friends want to start a company. One of the first things you'll do is sit down and think through what tasks must be done, who will do them, the order in which they will be done, and how they relate to each other. Ultimately, you might group these tasks into departments. Let's say you decide to have an order-taking department, an inventory department, and a shipping department. Each of your departments has its own unique tasks, keeping its staff members busy and requiring them to focus only on their own duties.

In this scenario, I'm using departments as a metaphor for the layers in a communication system. For things to run smoothly, the staff of each department has to trust and rely heavily on the others to do their jobs and competently handle their unique responsibilities. In your planning sessions, you'll probably take notes, recording the entire process to facilitate later discussions about standards of operation that will serve as your business blueprint or reference model.

Once your business is launched, your department heads, each armed with the part of the blueprint relating to their own department, will need to develop practical methods to implement their assigned tasks. These practical methods, or protocols, must be compiled into a standard-operating-procedures manual and followed closely. The procedures in your manual will have been included for different reasons and have varying degrees of importance and implementation. If you form a partnership or acquire another company, it will be imperative that its business protocols—its business blueprint—matches yours (or at least be compatible with it).

Similarly, software developers can use a reference model to understand computer communication processes and see what types of functions need to be accomplished on any one layer. If they're developing a protocol for a certain layer, all they need to concern themselves with is that specific layer's functions, not those of any other layer. Another layer and protocol will handle the other functions. The technical term for this idea is *binding*. The communication processes that are related to each other are bound, or grouped together, at a particular layer.

Advantages of Reference Models

The OSI model is hierarchical, and the same benefits and advantages can apply to any layered model. The primary purpose of all such models, especially the OSI model, is to allow different vendors' networks to interoperate.

Advantages of using the OSI layered model include, but are not limited to, the following:

- It divides the network communication process into smaller and simpler components, thus aiding component development, design, and troubleshooting.
- It allows multiple-vendor development through standardization of network components.

- It encourages industry standardization by defining what functions occur at each layer of the model.
- It allows various types of network hardware and software to communicate.
- It prevents changes in one layer from affecting other layers, so it doesn't hamper development and makes application programming easier.

The OSI Reference Model

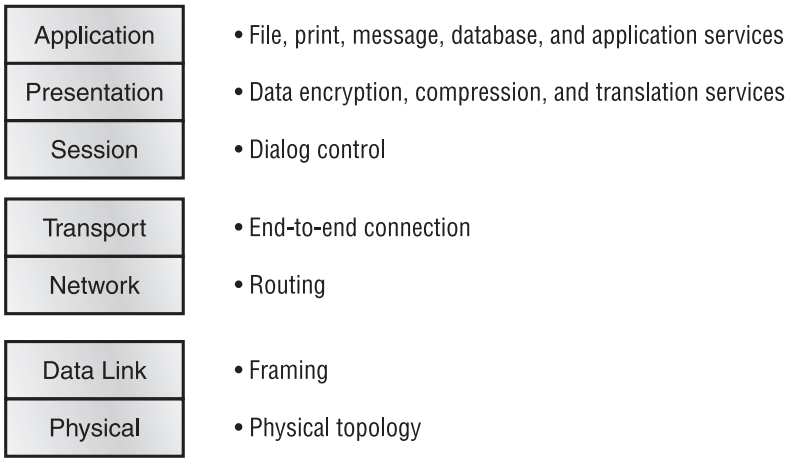
One of the greatest functions of the OSI specifications is to assist in data transfer between disparate hosts—meaning, for example, that they enable you to transfer data between a Unix host and a PC or a Mac.

The OSI model isn't a physical model, though. Rather, it's a set of guidelines that application developers can use to create and implement applications that run on a network. It also provides a framework for creating and implementing networking standards, devices, and internetworking schemes. The OSI model has seven layers:

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

Figure 2.1 shows a summary of the functions defined at each layer of the OSI model. With this in hand, you're now ready to explore each layer's function in detail.

FIGURE 2.1 Layer functions



Some people like to use the mnemonic Please Do Not Throw Sausage Pizza Away to remember the seven layers (starting at Layer 1 and moving up to Layer 7).

The seven layers are divided into two groups. The top three layers define how the applications within the end stations will communicate with each other and with users. The bottom four layers define how data is transmitted end to end. Figure 2.2 shows the three upper layers and their functions, and Figure 2.3 shows the four lower layers and their functions.

When you study Figure 2.2, understand that the user interfaces with the computer at the Application layer and also that the upper layers are responsible for applications communicating between hosts. Remember that none of the upper layers knows anything about networking or network addresses. That's the responsibility of the four bottom layers.

FIGURE 2.2 The upper layers

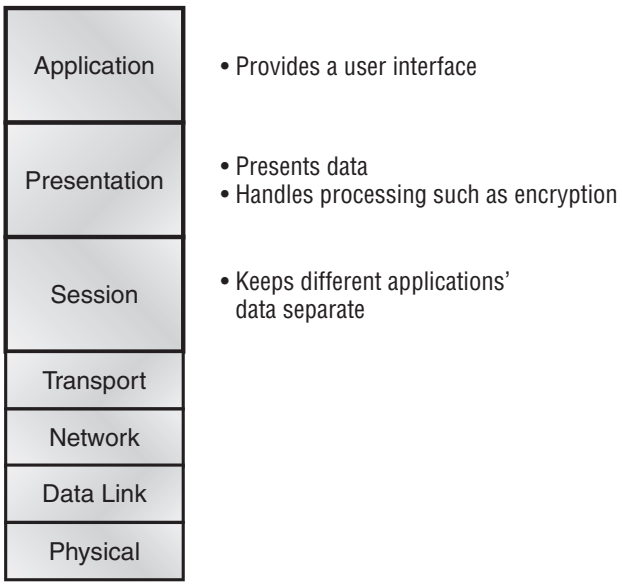


Figure 2.3 illustrates that the four bottom layers define how data is transferred through physical media, switches, and routers. These bottom layers also determine how to rebuild a data stream from a transmitting host to a destination host's application.

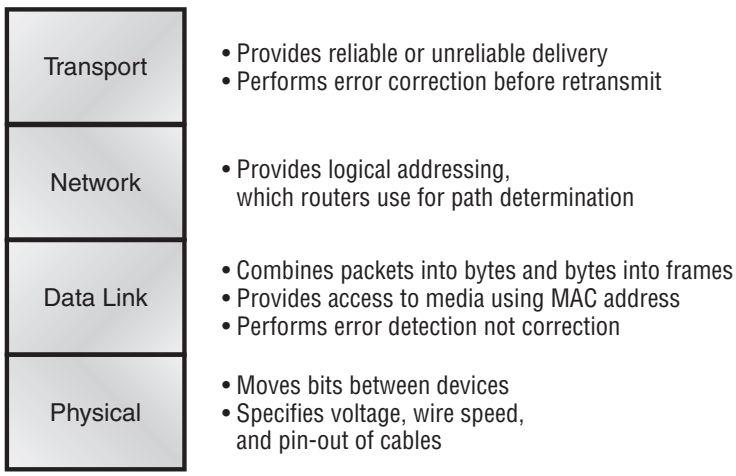
Let's start at the Application layer and work our way down the stack.

The Application Layer

The *Application layer* of the OSI model marks the spot where users actually communicate to the computer. This layer comes into play only when it's apparent that access to the network will be needed soon. Take the case of Internet Explorer (IE). You could uninstall every trace of networking components from a system, such as TCP/IP, the NIC card, and so on,

and you could still use IE to view a local HTML document—no problem. But things would definitely get messy if you tried to do something like view an HTML document that had to be retrieved using HTTP or nab a file with FTP or TFTP. That's because IE responds to requests such as those by attempting to access the Application layer. And what's happening is that the Application layer acts as an interface between the application program—which isn't part of the layered structure—and the next layer down by providing ways for the application to send information down through the protocol stack. In other words, IE doesn't reside within the Application layer—it interfaces with Application-layer protocols when it needs to deal with remote resources.

FIGURE 2.3 The lower layers



The Application layer is also responsible for identifying and establishing the availability of the intended communication partner and determining whether sufficient resources for the intended communication exist.

These tasks are important because computer applications sometimes require more than just desktop resources. Often, they unite communicating components from more than one network application. Prime examples are file transfers and email, as well as enabling remote access, network-management activities, client/server processes like printing, and information location. Many network applications provide services for communication over enterprise networks; but for present and future internetworking, the need is fast developing to reach beyond the limits of current physical networking.

It's important to remember that the Application layer acts as an interface between application programs. This means that Microsoft Word, for example, doesn't reside at the Application layer but instead interfaces with the Application-layer protocols. Chapter 5 will present some programs that actually reside at the Application layer—for example, FTP and TFTP.

The Presentation Layer

The *Presentation layer* gets its name from its purpose: It presents data to the Application layer and is responsible for data translation and code formatting.

This layer is essentially a translator and provides coding and conversion functions. A successful data-transfer technique is to adapt the data into a standard format before transmission. Computers are configured to receive this generically formatted data and then convert the data back into its native format for reading (for example, EBCDIC to ASCII). By providing translation services, the Presentation layer ensures that data transferred from one system's Application layer can be read by the Application layer of another one.

The OSI has protocol standards that define how standard data should be formatted. Tasks like data compression, decompression, encryption, and decryption are associated with this layer. In addition, some Presentation-layer standards are involved in multimedia operations.

The Session Layer

The *Session layer* is responsible for setting up, managing, and then tearing down sessions between Presentation-layer entities. This layer also provides dialogue control between devices, or nodes. It coordinates communication between systems and serves to organize their communication by offering three different modes: *simplex*, *half duplex*, and *full duplex*. To sum up, the Session layer basically keeps applications' data separate from other applications' data.

The Transport Layer

The *Transport layer* segments and reassembles data into a data stream. Services located in the Transport layer segment and reassemble data from upper-layer applications and unite it onto the same data stream. They provide end-to-end data transport services and can establish a logical connection between the sending host and destination host on an internetwork.

The Transport layer is responsible for providing mechanisms for multiplexing upper-layer applications, establishing sessions, and tearing down virtual circuits. It also hides details of any network-dependent information from the higher layers by providing transparent data transfer.

You may be familiar with TCP and UDP already. (If you're not, no worries—I'll tell you all about them in Chapter 6, "Introducing the Internet Protocol.") If so, you know that both work at the Transport layer and that TCP is a reliable service and UDP is not. This means application developers have more options because they have a choice between the two protocols when working with TCP/IP protocols.

The term *reliable networking* can be used at the Transport layer. It means that acknowledgments, sequencing, and flow control will be used.

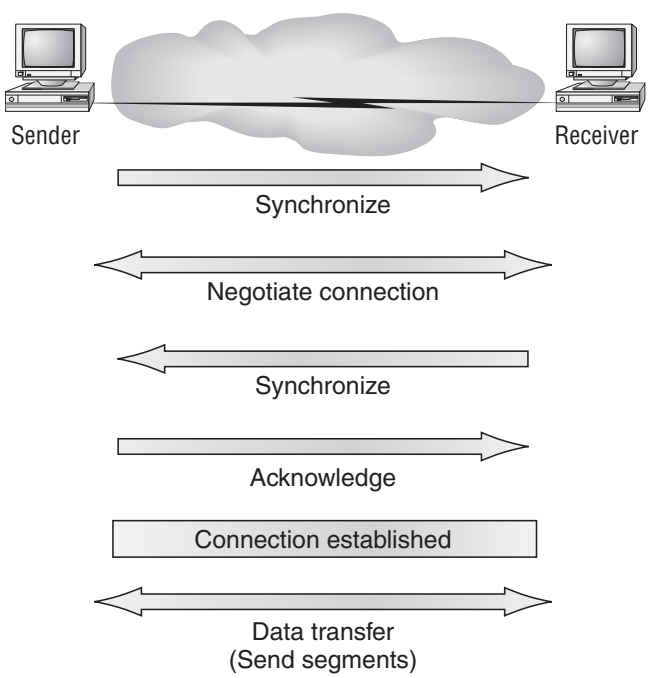
The Transport layer can be connectionless or connection-oriented. However, it's important that you understand the connection-oriented portion of the Transport layer. The following sections will provide the skinny on the connection-oriented (reliable) protocol of the Transport layer.

Connection-Oriented Communication

Before a transmitting host starts to send segments down the model, the sender's TCP process contacts the destination's TCP process to establish a connection. What is created is known as a *virtual circuit*. This type of communication is called *connection-oriented*. During this initial *handshake*, the two TCP processes also agree on the amount of information that will be sent in either direction before the respective recipient's TCP sends back an acknowledgment. With everything agreed on in advance, the path is paved for reliable communication to take place.

Figure 2.4 depicts a typical reliable session taking place between sending and receiving systems. Looking at it, you can see that both hosts' application programs begin by notifying their individual operating systems that a connection is about to be initiated. The two operating systems communicate by sending messages over the network confirming that the transfer is approved and that both sides are ready for it to take place. After all of this required synchronization takes place, a connection is fully established and the data transfer begins. This virtual circuit setup is called *overhead*.

FIGURE 2.4 Establishing a connection-oriented session



While the information is being transferred between hosts, the two machines periodically check in with each other, communicating through their protocol software to ensure that all is going well and that the data is being received properly.

Let me sum up the steps in the connection-oriented session—the three-way handshake—pictured in Figure 2.4:

- The first "connection agreement" segment is a request for synchronization.
- The second and third segments acknowledge the request and establish connection parameters—the rules—between hosts. These segments request that the receiver's sequencing is synchronized here as well so that a bidirectional connection is formed.
- The final segment is also an acknowledgment. It notifies the destination host that the connection agreement has been accepted and that the connection has been established. Data transfer can now begin.

Although I broke this connection setup into much detail, it really is just called the "three-way handshake" as I already mentioned, and is known as "SYN, SYN-ACK, SYN", or synchronize, synchronize-acknowledgment, syncronize.

Sounds pretty simple, but things don't always flow so smoothly. Sometimes congestion can occur during a transfer because a high-speed computer is generating data traffic a lot faster than the network can handle transferring it. A bunch of computers simultaneously sending datagrams through a single gateway or destination can also botch things up. In the latter case, a gateway or destination can become congested even though no single source caused the problem. In either case, the problem is basically akin to a freeway bottleneck—too much traffic for too small a capacity. It's not usually one car that's the problem; it's that there are simply too many cars on that particular freeway.

Flow Control

Data integrity is ensured at the Transport layer by maintaining *flow control* and by allowing users to request reliable data transport between systems. Flow control provides a means for the receiver to govern the amount of data sent by the sender. It prevents a sending host on one side of the connection from overflowing the buffers in the receiving host—an event that can result in lost data. Reliable data transport employs a connection-oriented communications session between systems, and the protocols involved ensure that the following will be achieved:

- The segments delivered are acknowledged back to the sender upon their reception.
- Any segments not acknowledged are retransmitted.
- Segments are sequenced back into their proper order upon arrival at their destination.
- A manageable data flow is maintained in order to avoid congestion, overloading, and data loss.

Okay, so what happens when a machine receives a flood of datagrams too quickly for it to process? It stores them in a memory section called a *buffer*. But this buffering action can solve the problem only if the datagrams are part of a small burst. If not, and the datagram deluge continues, a device's memory will eventually be exhausted, its flood capacity will be exceeded, and it will react by discarding any additional datagrams that arrive.

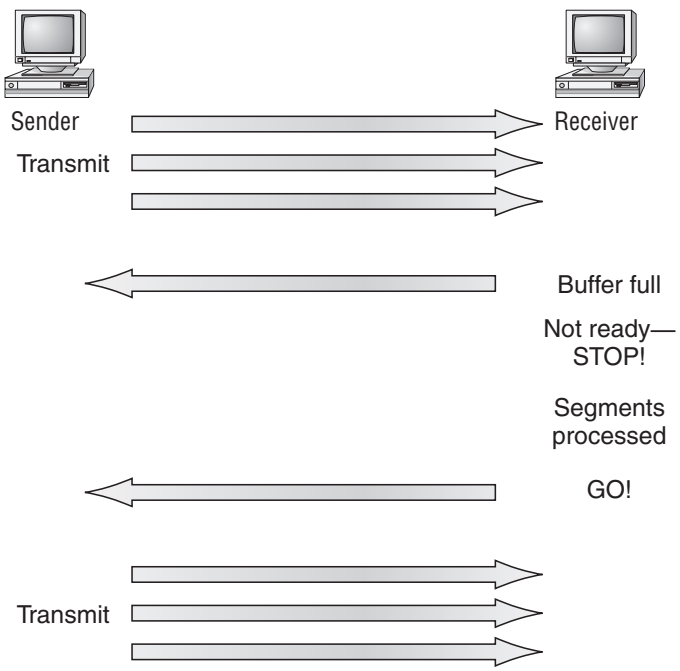
This sounds pretty bad, but there aren't really any huge worries here thanks to the transport function—network flood-control systems actually work quite well. How do they work? Well, instead of just dumping resources and allowing data to be lost, the transport can issue a "not ready" indicator to the sender, or source, of the flood (as shown in Figure 2.5). This mechanism works kind of like a stoplight, signaling the sending device to stop transmitting segment traffic to its overwhelmed peer. After the peer machine's receiver processes the segments abounding in its memory reservoir (its buffer), it sends out a "ready" transport indicator. When the machine waiting to transmit the rest of its datagrams receives this "go" indictor, it resumes its transmission.

During fundamental, reliable, connection-oriented data transfer, datagrams are delivered to the receiving host in exactly the same sequence they're transmitted—and the transmission fails if this order is breached! So if any data segments are lost, duplicated, or damaged along the way, a failure notice is transmitted. This problem is solved by making sure the receiving host acknowledges that it has received each and every data segment in the correct order.

To summarize, a service is considered connection-oriented if it has the following characteristics:

- A virtual circuit is set up (such as a three-way handshake).
- It uses sequencing.
- It uses acknowledgments.
- It uses flow control.

FIGURE 2.5 Transmitting segments with flow control



Windowing

Ideally, data throughput happens quickly and efficiently. And as you can imagine, it would be slow if the transmitting machine had to wait for an acknowledgment after sending each segment. But because time is available *after* the sender transmits the data segment and *before* it finishes processing acknowledgments from the receiving machine, the sender uses the break as an opportunity to transmit more data. The quantity of data segments (measured in bytes) that the transmitting machine is allowed to send without receiving an acknowledgment for them is called a *window*.

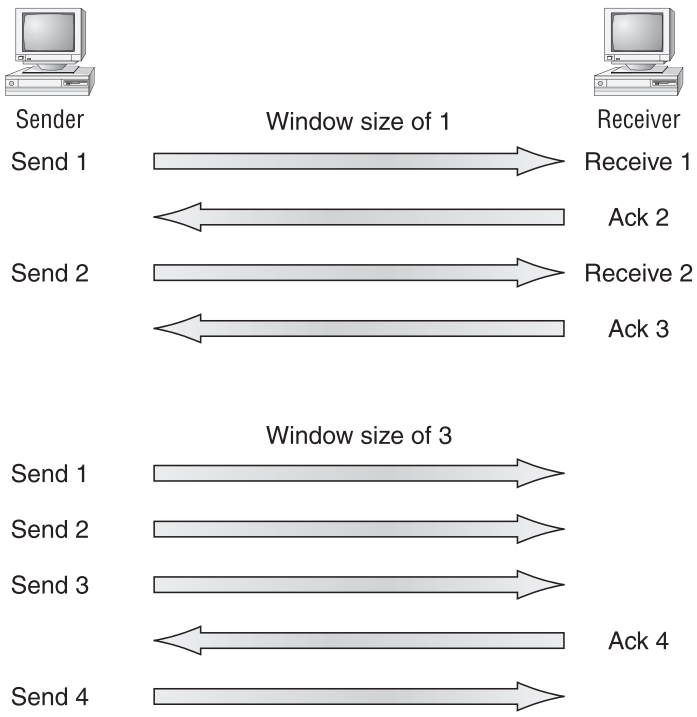
Windows are used to control the amount of outstanding, unacknowledged data segments.

It's important to understand that the size of the window controls how much information is transferred from one end to the other. Although some protocols quantify information by observing the number of packets, TCP/IP measures it by counting the number of bytes.

Figure 2.6 illustrates two window sizes—one set to 1 and one set to 3. In our simplified example, both the sending and receiving machines are workstations.

When you've configured a window size of 1, the sending machine waits for an acknowledgment for each data segment it transmits before transmitting another. If you've configured a window size of 3, the sending machine is allowed to transmit three data segments before an acknowledgment is received. In reality, this isn't done in simple numbers but in the amount of bytes that can be sent.

FIGURE 2.6 Windowing



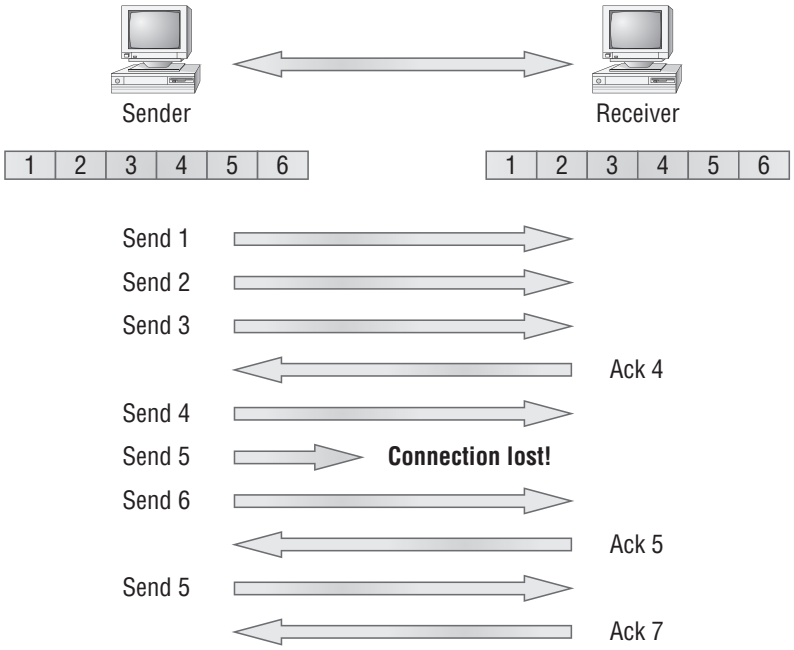
If a receiving host fails to receive all the segments that it should acknowledge, the host can improve the communication session by decreasing the window size.

Acknowledgments

Reliable data delivery ensures the integrity of a stream of data sent from one machine to the other through a fully functional data link. It guarantees that the data won't be duplicated or lost. This is achieved through something called *positive acknowledgment with retransmission*—a technique that requires a receiving machine to communicate with the transmitting source by sending an acknowledgment message back to the sender when it receives data. The sender documents each segment it sends and waits for this acknowledgment before sending the next segment. When it sends a segment, the transmitting machine starts a timer and retransmits if it expires before an acknowledgment is returned from the receiving end.

In Figure 2.7, the sending machine transmits segments 1, 2, and 3. The receiving node acknowledges it has received them by requesting segment 4. When it receives the acknowledgment, the sender then transmits segments 4, 5, and 6. If segment 5 doesn't make it to the destination, the receiving node acknowledges that event with a request for the segment to be re-sent. The sending machine will then resend the lost segment and wait for an acknowledgment, which it must receive in order to move on to the transmission of segment 7.

FIGURE 2.7 Transport layer reliable delivery



The Transport layer doesn't need to use a connection-oriented service (this is up to the application developer). It's safe to say that if you're connection-oriented, meaning that you've created a virtual circuit, you're using TCP. If you aren't setting up a virtual circuit, then you're using UDP and are considered connection-less.

Transport Control Protocol (TCP) and User Datagram Protocol (UDP) are protocols that work at the Transport layer and are covered in detail in Chapter 6.

Devices Used in an Internetwork

The following network devices operate at all seven layers of the OSI model:

- Network management stations (NMSs)
- Web and application servers
- Gateways (not default gateways)
- Network hosts

Several devices operate primarily at the Physical layer of the OSI model. These devices manipulate mainly the physical aspects of a network data stream (such as the voltages, signal direction, and signal strength). The most popular of these are the following:

- Network Interface Cards (NICs)
- Transceivers
- Repeaters
- Hubs

These devices are discussed in detail in Chapter 5, "Networking Devices."

The Network Layer

The *Network layer* manages device addressing, tracks the location of devices on the network, and determines the best way to move data, which means that the Network layer must transport traffic between devices that aren't locally attached. Routers (Layer 3 devices) are specified at the Network layer and provide the routing services within an internetwork.

It happens like this: First, when a packet is received on a router interface, the destination IP address is checked. If the packet isn't destined for that particular router, the router looks up the destination network address in the routing table. Once the router chooses an exit interface, the packet is sent to that interface to be framed and sent out on the local network. If the router can't find an entry for the packet's destination network in the routing table, the router drops the packet.

Two types of packets are used at the Network layer:

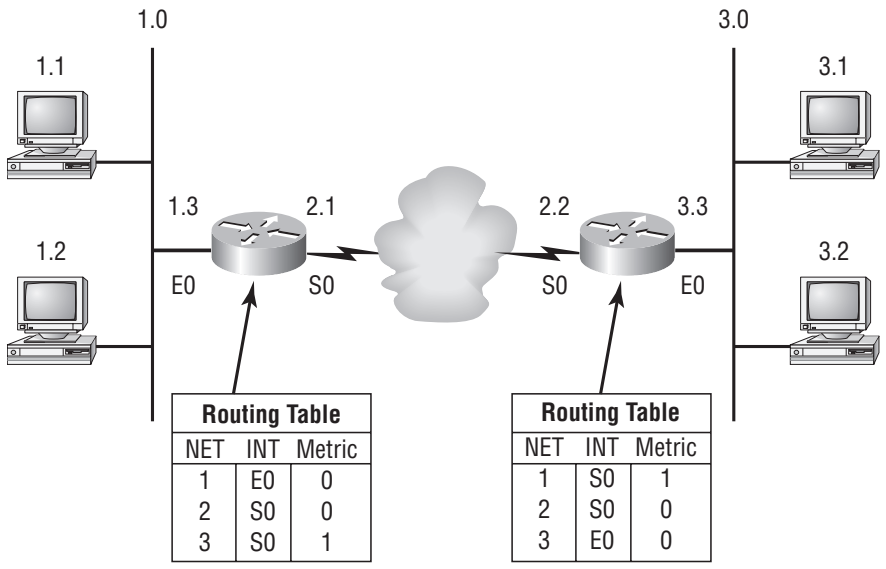
Data packets These are used to transport user data through the internetwork. Protocols used to support data traffic are called *routed protocols*. Two examples of routed protocols are Internet Protocol (IP) and Internet Protocol version 6 (IPv6), which you'll learn about in Chapter 7, "IP Addressing."

Route-update packets These are used to update neighboring routers about the networks connected to all routers within the internetwork. Protocols that send route-update packets are called routing protocols; examples of some common ones are Routing Information Protocol (RIP), RIPv2, Enhanced Interior Gateway Routing Protocol (EIGRP), and Open Shortest Path First (OSPF). Route-update packets are used to help build and maintain routing tables on each router.

Figure 2.8 shows an example of a routing table. The routing table used by a router includes the following information:

Network addresses These are protocol-specific network addresses. A router must maintain a routing table for individual routing protocols because each routing protocol keeps track of a network with a different addressing scheme (IP, and IPv6,for example). Think of it as a street sign in each of the different languages spoken by the residents who live on a particular street. If there were American, Spanish, and French folks on a street named Cat, the sign would read Cat/Gato/Chat.

FIGURE 2.8 Routing table used in a router



Interface This is the exit interface a packet will take when destined for a specific network.

Metric This value equals the distance to the remote network. Different routing protocols use different ways of computing this distance. I'll cover routing protocols in Chapter 9,

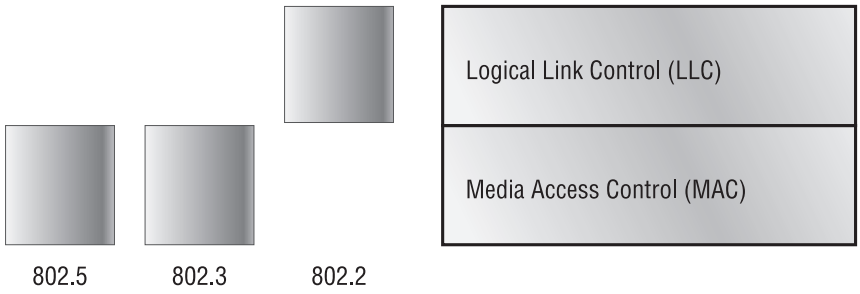
The Data Link Layer

The *Data Link layer* provides the physical transmission of the data and handles error notification, network topology, and flow control. This means the Data Link layer ensures that messages are delivered to the proper device on a LAN using hardware addresses, and translates messages from the Network layer into bits for the Physical layer to transmit.

The Data Link layer formats the message into pieces, each called a *data frame*, and adds a customized header containing the destination and source hardware address. This added information forms a sort of capsule that surrounds the original message in much the same way that engines, navigational devices, and other tools were attached to the lunar modules of the Apollo project. These various pieces of equipment were useful only during certain stages of flight and were stripped off the module and discarded when their designated stage was complete. This is a great analogy for data traveling through networks because it works very similarly.

Figure 2.10 shows the Data Link layer with the Ethernet and Institute of Electrical and Electronics Engineers (IEEE) specifications. When you check it out, notice that the IEEE 802.2 standard is not only used in conjunction with the other IEEE standards; it also adds functionality to those standards.

FIGURE 2.10 Data Link layer



It's important for you to understand that routers, which work at the Network layer, don't care about where a particular host is located. They're only concerned about where networks are located and the best way to reach them—including remote ones. Routers are totally obsessive when it comes to networks. And for once, this obsession is a good thing! The Data Link layer is responsible for the unique identification of each device that resides on a local network.

For a host to send packets to individual hosts on a local network as well as transmit packets between routers, the Data Link layer uses hardware addressing. Each time a packet is sent between routers, it's framed with control information at the Data Link layer; but that information is stripped off at the receiving router, and only the original packet is left completely intact. This framing of the packet continues for each hop until the packet is finally delivered to the correct receiving host. It's important to understand that the packet itself is never altered along the route; it's only encapsulated with the type of control information required for it to be properly passed on to the different media types.

The IEEE Ethernet Data Link layer has two sublayers:

Media Access Control (MAC) Defines how packets are placed on the media. Contention media access is "first come/first served" access where everyone shares the same bandwidth—hence the name. Physical addressing is defined here, as well as logical topologies. What's a logical topology? It's the signal path through a physical topology. Line discipline, error notification (not correction), ordered delivery of frames, and optional flow control can also be used at this sublayer.

Logical Link Control (LLC) Responsible for identifying Network-layer protocols and then encapsulating them. An LLC header tells the Data Link layer what to do with a packet once a frame is received. It works like this: A host receives a frame and looks in the LLC header to find out where the packet is destined—say, the IP protocol at the Network layer. The LLC can also provide flow control and sequencing of control bits.

Project 802

One of the major components of the Data Link layer is the result of the IEEE's 802 subcommittees and their work on standards for local area and metropolitan area networks (LANs/MANs). The committee met in February 1980, so they used the "80" from 1980 and the "2" from the second month to create the name Project 802. The designation for an 802 standard always includes a dot (.) followed by either a single or a double digit. These numeric digits specify particular categories within the 802 standard. These standards are listed in Table 2.1.

TABLE 2.1 IEEE 802 Networking Standards

Standard	Topic
802.1	LAN/MAN Management (and Media Access Control Bridges)
802.2	Logical Link Control
802.3	CSMA/CD
802.4	Token Passing Bus
802.5	Token Passing Ring
802.6	Distributed Queue Dual Bus (DQDB) Metropolitan Area Network (MAN)
802.7	Broadband Local Area Networks
802.8	Fiber-Optic LANs and MANs
802.9	Isochronous LANs

TABLE 2.1 IEEE 802 Networking Standards *(continued)*

Standard	Topic
802.10	LAN/MAN Security
802.11	Wireless LAN
802.12	Demand Priority Access Method
802.15	Wireless Personal Area Network
802.16	Wireless Metropolitain Area Network (also called WiMAX)
802.17	Resilient Packet Ring
802.18	LAN/MAN Standards Committee

The Physical Layer

Finally, we're hitting bottom. Well, not in a bad way—we've now arrived at the *Physical layer*, which does two important things: It sends bits and receives bits. Bits come only in values of 1 or 0—a Morse code with numerical values. The Physical layer communicates directly with the various types of actual communication media. Different kinds of media represent these bit values in different ways. Some use audio tones, and others employ *state transitions*—changes in voltage from high to low and low to high. Specific protocols are needed for each type of media to describe the proper bit patterns to be used, how data is encoded into media signals, and the various qualities of the physical media's attachment interface.

The Physical layer specifies the electrical, mechanical, procedural, and functional requirements for activating, maintaining, and deactivating a physical link between end systems. This layer is also where you identify the interface between the *data terminal equipment (DTE)* and the *data communication equipment (DCE)*. (Some older phone-company employees still call DCE data circuit-terminating equipment.) The DCE is usually located at the service provider, whereas the DTE is the attached device. The services available to the DTE are most often accessed via a modem or *channel service unit/data service unit (CSU/DSU)*.

The Physical layer's connectors and different physical topologies are defined by the OSI as standards, allowing disparate systems to communicate.

Finally, the Physical layer specifies the layout of the transmission media (its topology, in other words). A physical topology describes the way the cabling is physically laid out (as opposed to a logical topology, discussed earlier in the section "The Data Link Layer"). The physical topologies include Bus, Star, Ring, and Mesh, and were described in Chapter 1, "Introduction to Networks."

Introduction to Encapsulation

When a host transmits data across a network to another device, the data goes through *encapsulation*: It's wrapped with protocol information at each layer of the OSI model. Each layer communicates only with its peer layer on the receiving device.

To communicate and exchange information, each layer uses *Protocol Data Units (PDUs)*. These hold the control information attached to the data at each layer of the model. They're usually attached to the header in front of the data field but can also be in the trailer, or end, of it.

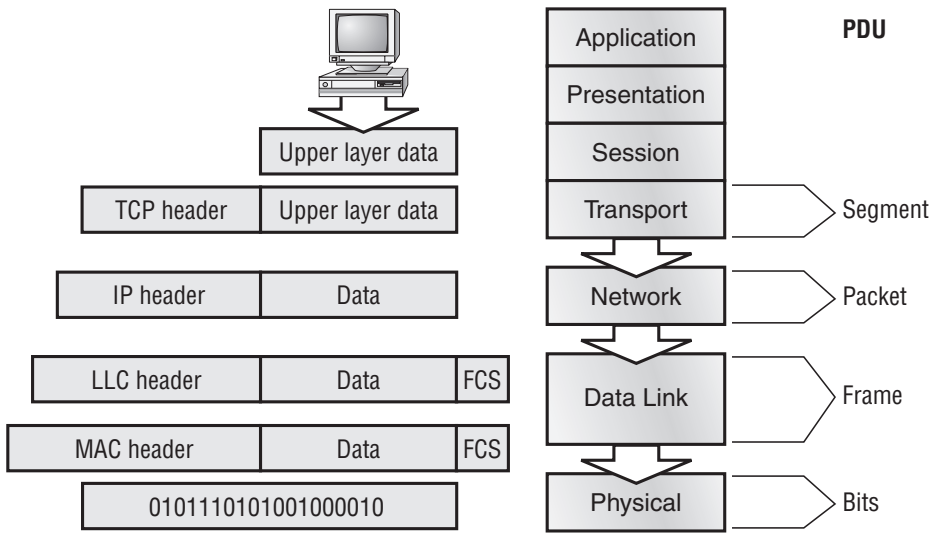
At a transmitting device, the data-encapsulation method works like this:

1. User information is converted to data for transmission on the network.
2. Data is converted to segments, and a reliable connection is set up between the transmitting and receiving hosts.
3. Segments are converted to packets or datagrams, and a logical address is placed in the header so each packet can be routed through an internetwork.
4. Packets or datagrams are converted to frames for transmission on the local network. Hardware (Ethernet) addresses are used to uniquely identify hosts on a local network segment.
5. Frames are converted to bits, and a digital encoding and clocking scheme is used.

Figure 2.11 shows how user data is encapsulated at a transmitting host.

After you learn more foundational material about networking in the next few chapters, I'll come back to the encapsulation method and discuss it in more detail in Chapter 6, as well as in even more detail in Chapter 9.

FIGURE 2.11: Data encapsulation



Summary

You're now armed with a ton of fundamental information. You're set to build on it and are well on your way to certification.

Let's take a minute to go over what you've learned in this chapter. We started by discussing internetworking models and the advantages of having them. I then discussed the OSI model—the seven-layer model used to help application developers design applications that can run on any type of system or network. Each layer has its special jobs and select responsibilities within the model to ensure that solid, effective communications do, in fact, occur. I provided you with complete details of each layer and discussed how you need to view the specifications of the OSI model.

This chapter finished with a brief introduction to the encapsulation method used in networking. Encapsulation is a highly important concept to understand, and I'll continue to discuss it throughout this book.

Exam Essentials

Remember the OSI layers. You absolutely must remember and understand the seven layers of the OSI model as well as what function each layer provides. The Application, Presentation, and Session layers are upper layers and are responsible for communicating from a user interface to an application. The Transport layer provides segmentation, sequencing, and virtual circuits. The Network layer provides logical network addressing and routing through an internetwork. The Data Link layer provides framing and placing of data on the network medium. The Physical layer is responsible for taking 1s and 0s and encoding them into a digital signal for transmission on the network segment.

Know the sublayers of the Data Link layer. In addition to the OSI layers, knowing the only layer that has sublayers and the functions of those sublayers is extremely important. The Data Link layer has two sublayers: LLC and MAC. The LLC sublayer is responsible primarily for the multiplexing of Network-layer protocols. The MAC sublayer is responsible for physical addressing and determining the appropriate time to place data on the network.

Know the devices that operate at each layer of the OSI model. Hubs and repeaters only see bits, making them Layer 1 devices. Because all networking devices have physical connectivity to the network, they all operate at Layer 1, but hubs and repeaters operate only at this layer. Nevertheless, we generally consider that a device operates at the highest layer it supports; that layer's functionality is the main reason we implement the device on the network. For example, switches and bridges are considered Layer 2 devices because they understand and make decisions based on Layer 2 addresses. Routers are Layer 3 devices for a similar reason; they deal with Layer 3 addresses. Networking devices, such as workstations, that run applications are said to operate at the Application layer (or you may hear that they operate at all layers) because they must include Application-layer protocols that offer services to networked applications.

Written Lab

1. Which layer chooses and determines the availability of communicating partners along with the resources necessary to make the connection, coordinates partnering applications, and forms a consensus on procedures for controlling data integrity and error recovery?
2. Which layer is responsible for converting data packets from the Data Link layer into electrical signals?
3. At which layer is routing implemented, enabling connections and path selection between two end systems?
4. Which layer defines how data is formatted, presented, encoded, and converted?
5. Which layer is responsible for creating, managing, and terminating sessions between applications?
6. Which layer manages the transmission of data across a physical link and is primarily concerned with physical addressing and the ordered delivery of frames?
7. Which layer is used for reliable communication between end nodes over the network and provides mechanisms for establishing, maintaining, and terminating virtual circuits as well as controlling the flow of information?
8. Which layer provides logical addressing that routers use for path determination?
9. Which layer specifies voltage, wire speed, and connector pinouts and moves bits between devices?
10. Which layer combines bits into bytes and bytes into frames and uses MAC addressing?

(The answers to the Written Lab can be found following the answers to the Review Questions for this chapter.)

Review Questions

1. Host 1 sent a SYN packet to Host 2. What will host 2 send in response?
   - **A.** A. ACK
   - **B.** NAK
   - **C.** SYN-ACK
   - **D.** SYN-NAK
   - **E.** SYN
2. TCP and UDP reside at which layer of the OSI model?
   - **A.** 1
   - **B.** 2
   - **C.** 3
   - **D.** 4
3. Which layer of the OSI model provides a user interface in the form of an entry point for programs to access the network infrastructure?
   - **A.** Application
   - **B.** Transport
   - **C.** Network
   - **D.** Physical
4. You are connected to a server on the Internet and you click on a link on the server and receive a time-out message. What layer could be the cause of this message?
   - **A.** Application
   - **B.** Transport
   - **C.** Network
   - **D.** Physical
5. Which layer of the OSI model is responsible for code and character-set conversion as well as recognizing data formats?
   - **A.** Application
   - **B.** Presentation
   - **C.** Session
   - **D.** Network

6. At which layers of the OSI model do bridges, hubs, and routers primarily operate, respectively?
   - **A.** Physical, Physical, Data Link
   - **B.** Data Link, Data Link, Network
   - **C.** Data Link, Physical, Network
   - **D.** Physical, Data Link, Network

7. Which layer of the OSI model is responsible for converting data into signals appropriate for the transmission medium?
   - **A.** Application
   - **B.** Network
   - **C.** Data Link
   - **D.** Physical

8. A receiving host has failed to receive all the segments that it should acknowledge. What can the host do to improve the reliability of this communication session?
   - **A.** Send a different source port number.
   - **B.** Restart the virtual circuit.
   - **C.** Decrease the sequence number.
   - **D.** Decrease the window size.

9. Which Layer 1 devices can be used to enlarge the area covered by a single LAN segment? (Choose two.)
   - **A.** Switch
   - **B.** NIC
   - **C.** Hub
   - **D.** Repeater
   - **E.** RJ-45 transceiver

10. Segmentation of a data stream happens at which layer of the OSI model?
    - **A.** Physical
    - **B.** Data Link
    - **C.** Network
    - **D.** Transport

11. When data is encapsulated, which is the correct order?
    - **A.** Data, frame, packet, segment, bits
    - **B.** Segment, data, packet, frame, bits
    - **C.** Data, segment, packet, frame, bits
    - **D.** Data, segment, frame, packet, bits

12. What are two purposes for segmentation with a bridge?

A. To add more broadcast domains

B. To create more collision domains

C. To add more bandwidth for users

D. To allow more broadcasts for users

13. Acknowledgments, sequencing, and flow control are characteristic of which OSI layer?

A. Layer 2

B. Layer 3

C. Layer 4

D. Layer 7

14. Which of the following are types of flow control? (Choose all that apply.)

A. Buffering

B. Cut-through

C. Windowing

D. Congestion avoidance

E. VLANs

15. What is the purpose of flow control?

A. To ensure that data is retransmitted if an acknowledgment is not received

B. To reassemble segments in the correct order at the destination device

C. To provide a means for the receiver to govern the amount of data sent by the sender

D. To regulate the size of each segment

16. At which layer of the OSI model would you find IP?

A. Transport

B. Network

C. Data Link

D. Physical

17. Of the following, which is the highest layer in the OSI model?

A. Transport

B. Session

C. Network

D. Presentation

18. Routers perform routing at which OSI layer?

- **A.** Physical
- **B.** Data Link
- **C.** Network
- **D.** Transport
- **E.** Application

19. Which of the following mnemonic devices can you use to remember the first letter of the name of each layer of the OSI model in the proper order?

- **A.** All People Seem To Need Processed Data
- **B.** Always Should People Never Threaten Dog Police
- **C.** Please Do Not Throw Sausage Pizza Away
- **D.** All Day People Should Try New Professions

20. Which IEEE standard specifies the protocol for CSMA/CD?

- **A.** 802.2
- **B.** 802.3
- **C.** 802.5
- **D.** 802.11

Answers to Review Questions

1. C. To set up a connection-oriented session, this is called a three-way handshake and the transmitting host sends a SYK packet, the receiving host sends a SYK-ACK, and the transmitting host replies with the last SYN packet. The session is now set up.
2. D. TCP and UDP are Transport-layer protocols. The Transport layer is Layer 4 of the OSI model.
3. A. The top layer of the OSI model gives applications access to the services that allow network access.
4. A. If the remote server is busy or does not respond to your web browser request, this is an Application layer problem.
5. B. The Presentation layer makes data "presentable" for the Application layer.
6. C. Bridges, like switches, are Data Link-layer devices. Hubs, like repeaters, are Physical-layer devices. Routers are Network-layer devices.
7. D. The Physical layer's job is to convert data into impulses that are designed for the wired or wireless medium being used on the attached segment.
8. D. A receiving host can control the transmitter by using flow control (TCP uses windowing by default). By decreasing the window size, the receiving host can slow down the transmitting host so the receiving host does not overflow its buffers.
9. C, D. Not that you really want to enlarge a single collision domain, but a hub (multiport repeater) will provide this functionality for you.
10. D. The Transport layer receives large data streams from the upper layers and breaks these up into smaller pieces called segments.
11. C. The encapsulation order is data, segment, packet, frame, bits.
12. B, C. Bridges and switches break up collision domains, which allow more bandwidth for users.
13. C. A reliable Transport-layer connection uses acknowledgments to make sure all data is received reliably. A reliable connection is defined by the use of acknowledgments, sequencing, and flow control, which is characteristic of the Transport layer (Layer 4).
14. A, C, D. The common types of flow control are buffering, windowing, and congestion avoidance.
15. C. Flow control allows the receiving device to control the pace of the transmitting device so the receiving device's buffer does not overflow.

16. B. IP is a Network-layer protocol. TCP is an example of a Transport-layer protocol, Ethernet is an example of a Data Link-layer protocol, and T1 can be considered a Physical-layer protocol.

17. D. The Presentation layer is the sixth layer of the model. Only the Application layer is higher, but it is not listed. Session is Layer 5, Transport is Layer 4, and Network is Layer 3.

18. C. A router is specified at the Network layer and a router routes packets. Routers can also be called layer-3 switches.

19. C. The phrase "Please Do Not Throw Sausage Pizza Away" contains the first letters of the layers in order from Layer 1 through Layer 7. "All People Seem To Need Data Processing" works from the top down, but that's not exactly how the option that looks similar reads. The other options have all the right letters, just not completely in the right order.

20. B. The 802.3 standard, commonly associated with Ethernet, specifies the media-access method used by Ethernet, which is known as Carrier Sense Multiple Access with Collision Detection (CSMA/CD).

Answers to Written Lab

1. The Application layer is responsible for finding the network resources broadcast from a server and adding flow control and error control (if the application developer chooses).
2. The Physical layer takes frames from the Data Link layer and encodes the 1s and 0s into a digital signal for transmission on the network medium.
3. The Network layer provides routing through an internetwork and logical addressing.
4. The Presentation layer makes sure that data is in a readable format for the Application layer.
5. The Session layer sets up, maintains, and terminates sessions between applications.
6. PDUs at the Data Link layer are called frames. As soon as you see *frame* in a question, you know the answer.
7. The Transport layer uses virtual circuits to create a reliable connection between two hosts.
8. The Network layer provides logical addressing, typically IP addressing, and routing.
9. The Physical layer is responsible for the electrical and mechanical connections between devices.
10. The Data Link layer is responsible for the framing of data packets.