This chapter covers the following subjects:

TCP/IP Networking Model: This section explains the terminology and concepts behind the world's most popular networking model, TCP/IP, including several example protocols: HTTP, TCP, IP, and Ethernet.

OSI Networking Model: This section explains the terminology behind the OSI networking model in comparison to TCP/IP.

CHAPTER 2

The TCP/IP and OSI Networking Models

You can think of a networking model as you think of a set of architectural plans for building a house. Sure, you can build a house without the architectural plans, but it will work better if you follow the plans. And because you probably have a lot of different people working on building your house, such as framers, electricians, bricklayers, painters, and so on, it helps if they can all reference the same plan. Similarly, you could build your own network, write your own software, build your own networking cards, and create a network without using any existing networking model. However, it is much easier to simply buy and use products that already conform to some well-known networking model. Because the networking product vendors use the same networking model, their products should work well together.

The CCNA exams include detailed coverage of one networking model: Transmission Control Protocol/Internet Protocol (TCP/IP). TCP/IP is the most pervasively used networking model in the history of networking. You can find support for TCP/IP on practically every computer operating system (OS) in existence today, from mobile phones to mainframe computers. Every network built using Cisco products today supports TCP/IP. And not surprisingly, the CCNA exams focus heavily on TCP/IP.

The ICND1 exam, and the ICND2 exam to a small extent, also covers a second networking model, called the Open System Interconnection (OSI) reference model. Historically, OSI was the first large effort to create a vendor-neutral networking model. Because of that timing, many of the terms used in networking today come from the OSI model, so this chapter's section on OSI discusses OSI and the related terminology.

"Do I Know This Already?" Quiz

The "Do I Know This Already?" quiz allows you to assess if you should read the entire chapter. If you miss no more than one of these ten self-assessment questions, you might want to move ahead to the "Exam Preparation Tasks" section. Table 2-1 lists the major headings in this chapter and the "Do I Know This Already?" quiz questions covering the material in those headings so you can assess your knowledge of these specific areas. The answers to the “Do I Know This Already?” quiz appear in Appendix A, “Answers to the ‘Do I Know This Already?’ Quizzes.”

Table 2-1 *Do I Know This Already?” Foundation Topics Section-to-Question Mapping*

Foundation Topics Section	Questions
TCP/IP Networking Model	1–6
OSI Networking Model	7–10

1. Which of the following protocols are examples of TCP/IP transport layer protocols? (Choose two answers.)

    a. Ethernet

    b. HTTP

    c. IP

    d. UDP

    e. SMTP

    f. TCP

2. Which of the following protocols are examples of TCP/IP network access layer protocols? (Choose two answers.)

    a. Ethernet

    b. HTTP

    c. IP

    d. UDP

    e. SMTP

    f. TCP

    g. PPP

3. The process of HTTP asking TCP to send some data and making sure that it is received correctly is an example of what?

    a. Same-layer interaction

    b. Adjacent-layer interaction

    c. OSI model

    d. All of these answers are correct.

4. The process of TCP on one computer marking a TCP segment as segment 1, and the receiving computer then acknowledging the receipt of TCP segment 1 is an example of what?

   a. Data encapsulation

   b. Same-layer interaction

   c. Adjacent-layer interaction

   d. OSI model

   e. All of these answers are correct.

5. The process of a web server adding a TCP header to the contents of a web page, followed by adding an IP header, and then adding a data link header and trailer is an example of what?

   a. Data encapsulation

   b. Same-layer interaction

   c. OSI model

   d. All of these answers are correct.

6. Which of the following terms is used specifically to identify the entity created when encapsulating data inside data link layer headers and trailers?

   a. Data

   b. Chunk

   c. Segment

   d. Frame

   e. Packet

   f. None of these—there is no encapsulation by the data link layer.

7. Which OSI layer defines the functions of logical network-wide addressing and routing?

   a. Layer 1

   b. Layer 2

   c. Layer 3

   d. Layer 4

   e. Layer 5

   f. Layer 6

   g. Layer 7

8. Which OSI layer defines the standards for cabling and connectors?

   a. Layer 1

   b. Layer 2

   c. Layer 3

   d. Layer 4

   e. Layer 5

   f. Layer 6

   g. Layer 7

9. Which OSI layer defines the standards for data formats and encryption?

   a. Layer 1

   b. Layer 2

   c. Layer 3

   d. Layer 4

   e. Layer 5

   f. Layer 6

   g. Layer 7

10. Which of the following terms are not valid terms for the names of the seven OSI layers? (Choose two answers.)

    a. Application

    b. Data link

    c. Transmission

    d. Presentation

    e. Internet

    f. Session

Foundation Topics

TCP/IP Networking Model

A *networking model*, sometimes also called either a *networking architecture* or *networking blueprint*, refers to a comprehensive set of documents. Individually, each document describes one small function required for a network; collectively, these documents define everything that should happen for a computer network to work. Some documents define a protocol, which is a set of logical rules that devices must follow to communicate. Other documents define some physical requirements for networking. For example, a document could define the voltage and current levels used on a particular cable when transmitting data.

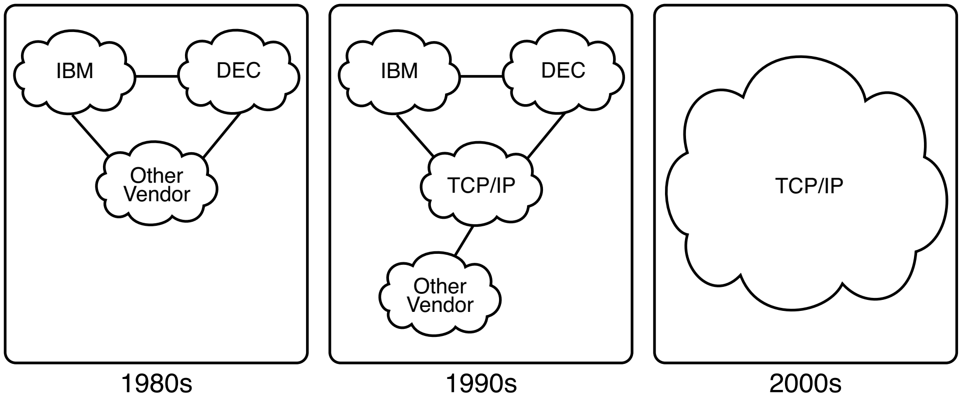
You can think of a networking model as you think of an architectural blueprint for building a house. Sure, you can build a house without the blueprint. However, the blueprint can ensure that the house has the right foundation and structure so it will not fall down, and it has the correct hidden spaces to accommodate the plumbing, electrical, gas, and so on. Also, the many different people that build the house using the blueprint—such as framers, electricians, bricklayers, painters, and so on—know that if they follow the blueprint, their part of the work should not cause problems for the other workers.

Similarly, you could build your own network—write your own software, build your own networking cards, and so on—to create a network. However, it is much easier to simply buy and use products that already conform to some well-known networking model or blueprint. Because the networking product vendors build their products with some networking model in mind, their products should work well together.

History Leading to TCP/IP

Today, the world of computer networking uses one networking model: TCP/IP (Transmission Control Protocol / Internet Protocol). However, the world has not always been so simple. Once upon a time, there were no networking protocols, including TCP/IP. Vendors created the first networking protocols; these protocols supported only that vendor's computers. For instance, IBM published its Systems Network Architecture (SNA) networking model in 1974. Other vendors also created their own proprietary networking models. As a result, if your company bought computers from three vendors, network engineers often had to create three different networks based on the networking models created by each company, and then somehow connect those networks, making the combined networks much more complex. The left side of Figure 2-1 shows the general idea of what a company's enterprise network might have looked back in the 1980s, before TCP/IP became common in enterprise internetworks.

Figure 2-1 *Historical Progression: Proprietary Models to the Open TCP/IP Model*



Although vendor-defined proprietary networking models often worked well, having an open, vendor-neutral networking model would aid competition and reduce complexity. The International Organization for Standardization (ISO) took on the task to create such a model, starting as early as the late 1970s, beginning work on what would become known as the Open System Interconnection (OSI) networking model. ISO had a noble goal for the OSI model: to standardize data networking protocols to allow communication between all computers across the entire planet. ISO worked toward this ambitious and noble goal, with participants from most of the technologically developed nations on Earth participating in the process.

A second, less formal effort to create an open, vendor-neutral, public networking model sprouted forth from a U.S. Department of Defense (DoD) contract. Researchers at various universities volunteered to help further develop the protocols surrounding the original DoD work. These efforts resulted in a competing open networking model called TCP/IP.

During the 1990s, companies began adding OSI, TCP/IP, or both to their enterprise networks. However, by the end of the 1990s, TCP/IP had become the common choice, and OSI fell away. The center part of Figure 2-1 shows the general idea behind enterprise networks in that decade—still with networks built upon multiple networking models, but including TCP/IP.

Here in the 21st century, TCP/IP dominates. Proprietary networking models still exist, but they have mostly been discarded in favor of TCP/IP. The OSI model, whose development suffered in part because of a slower formal standardization process as compared with TCP/IP, never succeeded in the marketplace. And TCP/IP, the networking model originally created almost entirely by a bunch of volunteers, has become the most prolific network model ever, as shown on the right side of Figure 2-1.

In this chapter, you will read about some of the basics of TCP/IP. Although you will learn some interesting facts about TCP/IP, the true goal of this chapter is to help you understand what a networking model or networking architecture really is and how it works.

Also in this chapter, you will learn about some of the jargon used with OSI. Will any of you ever work on a computer that is using the full OSI protocols instead of TCP/IP? Probably not. However, you will often use terms relating to OSI. Also, the ICND1 exam covers the basics of OSI, so this chapter also covers OSI to prepare you for questions about it on the exam.

Overview of the TCP/IP Networking Model

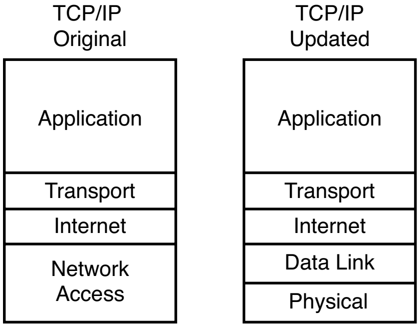
The TCP/IP model both defines and references a large collection of protocols that allow computers to communicate. To define a protocol, TCP/IP uses documents called Requests for Comments (RFC). (You can find these RFCs using any online search engine.) The TCP/IP model also avoids repeating work already done by some other standards body or vendor consortium by simply referring to standards or protocols created by those groups. For example, the Institute of Electrical and Electronic Engineers (IEEE) defines Ethernet LANs; the TCP/IP model does not define Ethernet in RFCs, but refers to IEEE Ethernet as an option.

An easy comparison can be made between telephones and computers that use TCP/IP. You go to the store and buy a phone from one of a dozen different vendors. When you get home and plug in the phone to the same cable in which your old phone was connected, the new phone works. The phone vendors know the standards for phones in their country and build their phones to match those standards.

Similarly, when you buy a new computer today, it implements the TCP/IP model to the point that you can usually take the computer out of the box, plug in all the right cables, turn it on, and it connects to the network. You can use a web browser to connect to your favorite website. How? Well, the OS on the computer implements parts of the TCP/IP model. The Ethernet card, or wireless LAN card, built into the computer implements some LAN standards referenced by the TCP/IP model. In short, the vendors that created the hardware and software implemented TCP/IP.

To help people understand a networking model, each model breaks the functions into a small number of categories called *layers*. Each layer includes protocols and standards that relate to that category of functions. TCP/IP actually has two alternative models, as shown in Figure 2-2.

Figure 2-2 *The Two TCP/IP Networking Models*



The model on the left, the original TCP/IP model, breaks TCP/IP into four layers. The top layers focus more on the applications that need to send and receive data, whereas the lower layers focus more on the need to somehow transmit the bits from one device to another. The model on the right is a newer version of the model, formed by expanding the network access layer on the left into two separate layers: data link and physical. Note that the model on the right is used more often today.

Many of you will have already heard of several TCP/IP protocols, like the examples listed in Table 2-2. Most of the protocols and standards in this table will be explained in more detail as you work through this book. Following the table, this section takes a closer look at the layers of the TCP/IP model.

Table 2-2 *TCP/IP Architectural Model and Example Protocols*

TCP/IP Architecture Layer	Example Protocols
Application	HTTP, POP3, SMTP
Transport	TCP, UDP
Internet	IP
Network Access	Ethernet, Point-to-Point Protocol (PPP), T/1

TCP/IP Application Layer

TCP/IP application layer protocols provide services to the application software running on a computer. The application layer does not define the application itself, but it defines services that applications need. For example, application protocol HTTP defines how web browsers can pull the contents of a web page from a web server. In short, the application layer provides an interface between software running on a computer and the network itself.

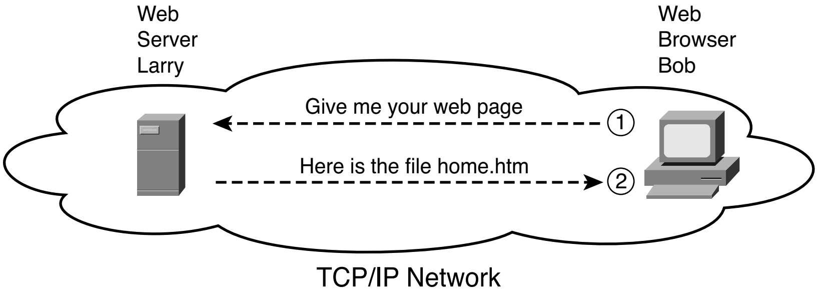
Arguably, the most popular TCP/IP application today is the web browser. Many major software vendors either have already changed or are changing their application software to support access from a web browser. And thankfully, using a web browser is easy: you start a web browser on your computer and select a website by typing the name of the website, and the web page appears.

HTTP Overview

What really happens to allow that web page to appear on your web browser?

Imagine that Bob opens his browser. His browser has been configured to automatically ask for web server Larry's default web page, or *home page*. The general logic looks like Figure 2-3.

Figure 2-3 *Basic Application Logic to Get a Web Page*



So, what really happened? Bob's initial request actually asks Larry to send his home page back to Bob. Larry's web server software has been configured to know that the default web page is contained in a file called home.htm. Bob receives the file from Larry and displays the contents of the file in the web-browser window.

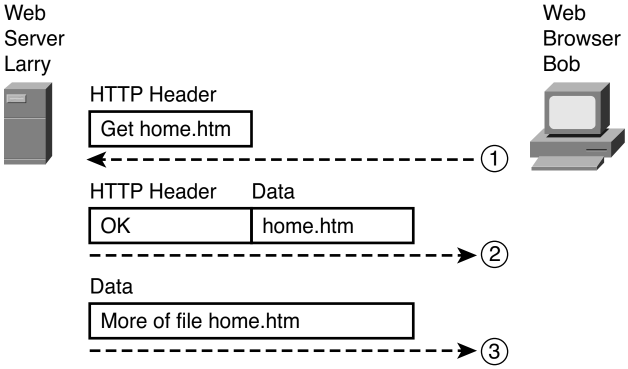
HTTP Protocol Mechanisms

Taking a closer look, this example shows how applications on each endpoint computer—specifically, the web-browser application and web-server application—use a TCP/IP application layer protocol. To make the request for a web page and return the contents of the web page, the applications use the Hypertext Transfer Protocol (HTTP).

HTTP did not exist until the Tim Berners-Lee created the first web browser and web server in the early 1990s. Berners-Lee gave HTTP functions to ask for the contents of web pages, specifically by giving the web browser the ability to request files from the server, and giving the server a way to return the content of those files. The overall logic matches what was shown in Figure 2-3; Figure 2-4 shows the same idea, but with details specific to HTTP.

AUTHOR'S NOTE The full version of most web addresses—also called universal resource locators (URL)—begin with the letters "http," which means that HTTP is used to transfer the web pages.

Figure 2-4 *HTTP Get Request, HTTP Reply, and One Data-Only Message*



To get the web page from Larry, at Step 1, Bob sends a message with an HTTP header. Generally, protocols use headers as a place to put information used by that protocol. This HTTP header includes the request to "get" a file. The request typically contains the name of the file (home.htm, in this case), or, if no filename is mentioned, the web server assumes that Bob wants the default web page.

Step 2 in Figure 2-4 shows the response from web server Larry. The message begins with an HTTP header, with a return code (200), which means something as simple as "OK" returned in the header. HTTP also defines other return codes, so the server can tell the browser whether the request worked or not. (As another example: If you ever looked for a web page that was not found, and then received an HTTP 404 "not found" error, you received an HTTP return code of 404.) The second message also includes the first part of the requested file.

Step 3 in Figure 2-4 shows another message from web server Larry to web browser Bob, but this time without an HTTP header. HTTP transfers the data by sending multiple messages, each with a part of the file. Rather than wasting space by sending repeated HTTP headers that list the same information, these additional messages simply omit the header.

TCP/IP Transport Layer

Although many TCP/IP application layer protocols exist, the TCP/IP transport layer includes a smaller number of protocols. The two most commonly used transport layer protocols are the *Transmission Control Protocol* (TCP) and the *User Datagram Protocol* (UDP).

Transport layer protocols provide services to the application layer protocols that reside one layer higher in the TCP/IP model. How does a transport layer protocol provide a service to a higher layer protocol? This section introduces that general concept by focusing on a single service provided by TCP: error recovery. Later chapters examine the transport layer in more detail, and discuss more functions of the transport layer.

TCP Error Recovery Basics

To appreciate what the transport layer protocols do, you must think about the layer above the transport layer, the application layer. Why? Well, each layer provides a service to the layer above it, like the error-recovery service provided to application layer protocols by TCP.

For example, in Figure 2-3, Bob and Larry used HTTP to transfer the home page from web server Larry to Bob's web browser. But what would have happened if Bob's HTTP GET request had been lost in transit through the TCP/IP network? Or, what would have happened if Larry's response, which included the contents of the home page, had been lost? Well, as you might expect, in either case, the page would not have shown up in Bob's browser.

TCP/IP needs a mechanism to guarantee delivery of data across a network. Because many application layer protocols probably want a way to guarantee delivery of data across a network, the creators of TCP included an error recovery feature. To recover from errors, TCP uses the concept of acknowledgments. Figure 2-5 outlines the basic idea behind how TCP notices lost data and asks the sender to try again.

Figure 2-5 *TCP Error Recovery Services as Provided to HTTP*

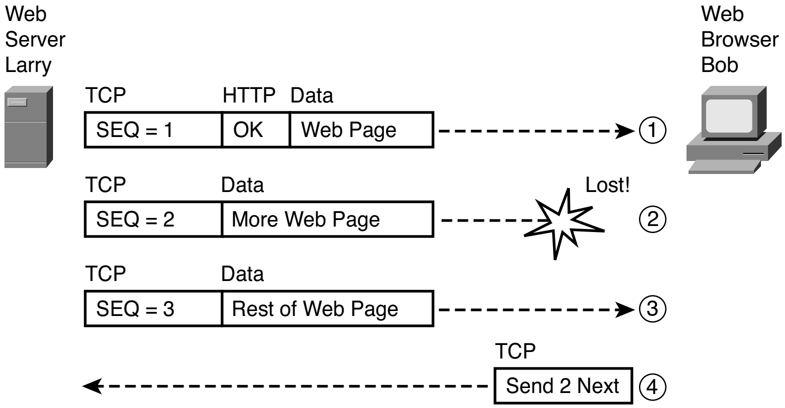


Figure 2-5 shows web server Larry sending a web page to web browser Bob, using three separate messages. Note that this figure shows the same HTTP headers as Figure 2-4, but it also shows a TCP header. The TCP header shows a sequence number (SEQ) with each message. In this example, the network has some problem so that the network fails to deliver the segment with sequence number 2. When Bob receives messages with sequence numbers 1 and 3, but does not receive a message with sequence number 2, Bob realizes that message 2 was lost. That realization by Bob's TCP logic causes Bob to send a TCP segment back to Larry, asking Larry to send message 2 again.

Same Layer and Adjacent Layer Interactions

The example in Figure 2-4 also demonstrates a function called *adjacent-layer interaction*, which refers to the concepts of how adjacent layers in a networking model, on the same computer, work together. In this example, the higher-layer protocol (HTTP) needs to do something it cannot do (error recovery). The higher layer asks for the next lower-layer protocol (TCP) to perform the service; the lower layer provides a service to the layer above it.

Figure 2-4 also shows an example of a similar function called *same-layer interaction.* When a particular layer on one computer wants to communicate with the same layer on another computer, the two computers use headers to hold the information that they want to communicate. For example, in Figure 2-4, Larry set the sequence numbers to 1, 2, and 3, so that Bob could notice when some of the data did not arrive. Larry's TCP process created that TCP header with the sequence number; Bob's TCP process received and reacted to the TCP segments. This process through which two computers set and interpret the information in the header used by that layer is called *same-layer interaction*, and it occurs between different computers.

Table 2-3 summarizes the key points about how adjacent layers work together on a single computer and how one layer on one computer works with the same networking layer on another computer

Key Topic

Table 2-3 *Summary: Same-Layer and Adjacent-Layer Interactions*

Concept	**Description**
Same-layer interaction on different computers	The two computers use a protocol to communicate with the same layer on another computer. The protocol defined by each layer uses a header that is transmitted between the computers to communicate what each computer wants to do.
Adjacent-layer interaction on the same computer	On a single computer, one layer provides a service to a higher layer. The software or hardware that implements the higher layer requests that the next lower layer perform the needed function.

TCP/IP Internet Layer

The application layer includes many protocols. The transport layer includes fewer, most notably, TCP and UDP. The TCP/IP Internet layer includes a small number of protocols, but only one major protocol: the *Internet Protocol* (IP). In fact, the name *TCP/IP* is simply the names of the two most common protocols (TCP and IP) separated by a /.

IP provides several features, most importantly, addressing and routing. This section begins by comparing IP's addressing and routing with another commonly known system that uses addressing and routing: the postal service. Following that, this section introduces IP addressing and routing. (More details follow in Chapter 5, "Fundamentals of IPv4 Addressing and Routing.")

Internet Protocol and the Postal Service

Imagine that you just wrote two letters: one to a friend on the other side of the country and one to a friend on the other side of town. You addressed the envelopes and put on the stamps, so both are ready to give to the postal service. Is there much difference in how you treat each letter? Not really. Typically, you would just put them in the same mailbox, and expect the postal service to deliver both letters.

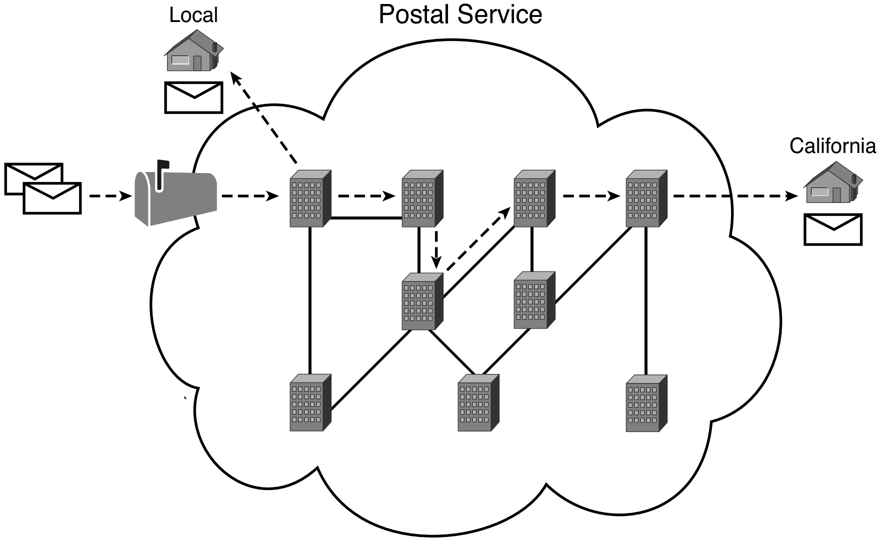
The postal service, however, must think about each letter separately, and then make a decision of where to send each letter so it is delivered. For the letter sent across town, the people in the local post office probably just need to put the letter on another truck.

For the letter that needs to go across the country, the postal service sends the letter to another post office, then another, and so on, until the letter gets delivered across the country. At each post office, the postal service must process the letter and choose where to send it next.

To make it all work, the postal service has regular routes for small trucks, large trucks, planes, boats, and so on, to move letters between postal service sites. The service must be able to receive and forward the letters, and it must make good decisions about where to send each letter next, as shown in Figure 2-6.

Still thinking about the postal service, consider the difference between the person sending the letter and the work that the postal service does. The person sending the letters expects that the postal service will deliver the letter most of the time. However, the person sending the letter does not need to know the details of exactly what path the letters take. In contrast, the postal service does not create the letter, but they accept the letter from the customer. Then, the postal service must know the details about addresses, postal codes that group addresses into larger groups, and it must have the ability to deliver the letters.

Figure 2-6 *Postal Service Forwarding (Routing) Letters*



The TCP/IP application and transport layers act like the person sending letters through the postal service. These upper layers work the same way regardless of whether the endpoint host computers are on the same LAN or are separated by the entire Internet. To send a message, these upper layers ask the layer below them, the Internet layer, to deliver the message.

The lower layers of the TCP/IP model, the Internet layer and the network access layer, act more like the postal service to deliver those messages to the correct destinations. To do so, these lower layers must understand the underlying physical network because they must choose how to best deliver the data from one host to another.

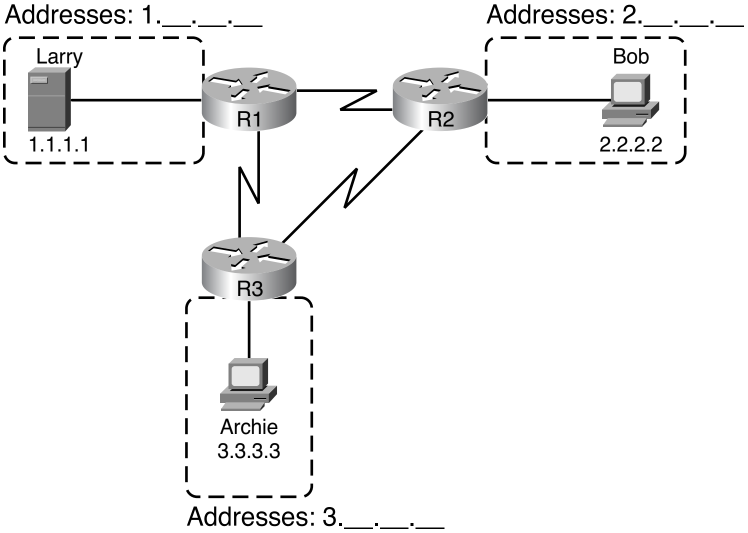
So, what does this all matter to networking? Well, the Internet layer of the TCP/IP networking model, primarily defined by the *Internet Protocol* (IP), works much like the postal service. IP defines addresses so that each host computer can have a different IP address, just as the postal service defines addressing that allows unique addresses for each house, apartment, and business. Similarly, IP defines the process of routing so that devices called routers can work like the post office to forward packets of data so that they are delivered to the correct destinations. Just as the postal service created the necessary infrastructure to be able to deliver letters—post offices, sorting machines, trucks, planes, and personnel—the Internet layer defines the details of how a network infrastructure should be created so that the network can deliver data to all computers in the network.

Internet Protocol Addressing Basics

IP defines addresses for several important reasons. First, each device that uses TCP/IP—each TCP/IP *host*—needs a unique address so that it can be identified in the network. IP also defines how to group addresses together, just like the postal system groups addresses based on postal codes (like ZIP codes in the U.S.).

To understand the basics, examine Figure 2-7, which shows the familiar web server Larry and web browser Bob; but now, instead of ignoring the network between these two computers, part of the network infrastructure is included.

Figure 2-7 *Simple TCP/IP Network: Three Routers with IP Addresses Grouped*



First, note that Figure 2-7 shows some sample IP addresses. Each IP address has four numbers, separated by periods. In this case, Larry uses IP address 1.1.1.1, and Bob uses 2.2.2.2. This style of number is called a *dotted-decimal notation* (DDN).

Figure 2-7 also shows three groups of address. In this example, all IP address that begin with 1 must be on the upper left, as shown in shorthand in the figure as 1._._._. All addresses that begin with 2 must be on the right, as shown in shorthand as 2._._._. Finally, all IP addresses that begin with 3 must be on the bottom of the figure.

Additionally, Figure 2-7 also introduces icons that represent IP routers. Routers are networking devices that connect the parts of the TCP/IP network together for the purpose of routing (forwarding) IP packets to the correct destination. Routers do the equivalent of the work done by each post office site: they receive IP packets on various physical interfaces, make decisions based on the IP address included with the packet, and then physically forward the packet out some other network interface.

IP Routing Basics

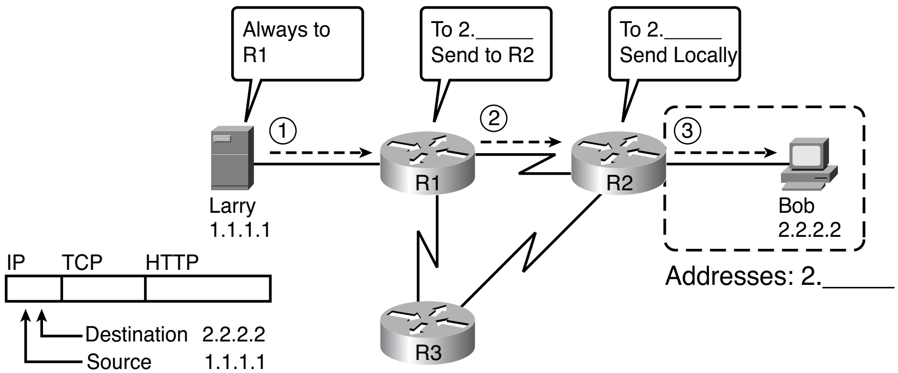
The TCP/IP Internet layer, using the IP protocol, provides a service of forwarding IP packets from one device to another. Any device with an IP address can connect to the TCP/IP network and send packets. This section shows a basic IP routing example for perspective.

NOTE The term *IP host* refers to any device, regardless of size or power, that has an IP address and connects to any TCP/IP network.

Figure 2-8 repeats the familiar case in which web server Larry wants to send part of a web page to Bob, but now with details related to IP. On the lower left, note that server Larry has the familiar application data, HTTP header, and TCP header ready to send. Additionally, the message now also contains an IP header. The IP header includes a source IP address of Larry's IP (1.1.1.1) and a destination IP address of Bob's IP address (2.2.2.2).

Key Topic

Figure 2-8 *Basic Routing Example*



Step 1, on the left of Figure 2-8, begins with Larry being ready to send an IP packet. Larry's IP process chooses to send the packet to some router—a nearby router on the same LAN—with the expectation that the router will know how to forward the packet. (This logic is much like you or me sending all of our letters by putting them in a nearby post office box.) Larry doesn't need to know anything more about the topology or the other routers.

At Step 2, router R1 receives the IP packet, and R1's IP process makes a decision. R1 looks at the destination address (2.2.2.2), compares that address to its known IP routes, and chooses to forward the packet to router R2. This process of forwarding the IP packet is called *IP routing* (or simply *routing*).

At Step 3, router R2 repeats the same kind of logic used by router R1. R2's IP process will compare the packet's destination IP address (2.2.2.2) to R2's known IP routes and make a choice to forward the packet to the right, on to Bob.

All the CCNA exams cover IP fairly deeply. Practically half the chapters in this book discuss some feature that relates to addressing, IP routing, and how routers perform routing.

TCP/IP Network Access Layer

The TCP/IP model's network access layer defines the protocols and hardware required to deliver data across some physical network. The term *network access* refers to the fact that this layer defines how to access or use the physical media over which data can be transmitted.

Just like every layer in any networking model, the TCP/IP network access layer provides services to the layer above it in the model. When a host or router's IP process chooses to send an IP packet to another router or host, that host or router then uses network access layer details to send that packet to the next host/router.

Because each layer provides a service to the layer above it, take a moment to think about the IP logic related to Figure 2-8. In that example, host Larry's IP logic chooses to send the IP packet to a nearby router (R1), with no mention of the underlying Ethernet. The Ethernet network, which implements access layer protocols, must then be used to deliver that packet from host Larry over to router R1. Figure 2-9 shows four steps of what occurs at the network access layer to allow Larry to send the IP packet to R1.

NOTE Figure 2-9 depicts the Ethernet as a series of lines. Networking diagrams often use this convention when drawing Ethernet LANs, in cases where the actual LAN cabling and LAN devices are not important to some discussion, as is the case here. The LAN would have cables and devices, like LAN switches, which are not shown in this figure.

Figure 2-9 *Larry Using Ethernet to Forward an IP Packet to Router R1*

Key Topic

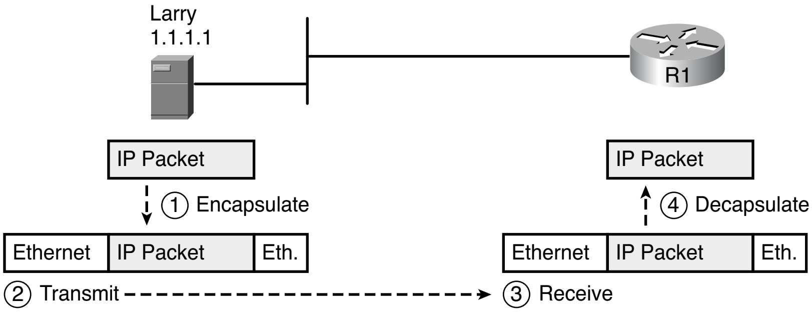


Figure 2-9 shows four steps. The first two occur on Larry, and the last two occur on router R1, as follows:

Step 1 Larry encapsulates the IP packet between an Ethernet header and Ethernet trailer, creating an Ethernet *frame*.

Step 2 Larry physically transmits the bits of this Ethernet frame, using electricity flowing over the Ethernet cabling.

Step 3 Router R1 physically receives the electrical signal over a cable, and re-creates the same bits by interpreting the meaning of the electrical signals.

Step 4 Router R1 de-encapsulates the IP packet from the Ethernet frame by removing and discarding the Ethernet header and trailer.

By the end of this process, the network access processes on Larry and R1 have worked together to deliver the packet from Larry to router R1.

NOTE Protocols define both headers and trailers for the same general reason, but headers exist at the beginning of the message, and trailers exist at the end.

The network access layer includes a large number of protocols and standards. For instance, the network access layer includes all the variations of Ethernet protocols, along with several other LAN standards that were more popular in decades past. The network access layer includes WAN standards for different physical media, which differ significantly compared to LAN standards because of the longer distances involved in transmitting the data. This layer also includes the popular WAN standards that add headers and trailers as shown generally in Figure 2-7, protocols such as the Point-to-Point Protocol (PPP) and Frame Relay. Chapter 3, “Fundamentals of LANs,” and Chapter 4, “Fundamentals of WANs,” further develop these topics for LAN and WAN, respectively.

In short, the TCP/IP network access layer includes two distinct functions: functions related to the physical transmission of the data, plus the protocols and rules that control the use of the physical media. The five layer TCP/IP model simply splits out the network access layer into two layers (Data Link and Physical) to match this logic.

TCP/IP Model and Terminology

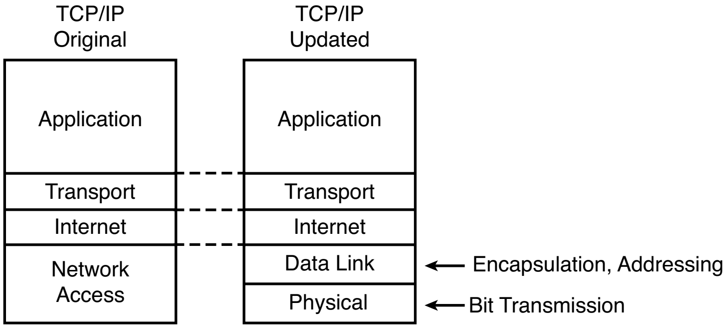
Before completing this introduction to the TCP/IP model, this section examines a few remaining details of the model and some related terminology.

Comparing the Two TCP/IP Models

The functions defined in the network access layer can be broken into two major categories: functions related directly to the physical transmission of data and those only indirectly related to the physical transmission of data. For instance, in the four steps shown around Figure 2-9, Steps 2 and 3 were specific to sending the data, but Steps 1 and 4—encapsulation and de-encapsulation—were only indirectly related. This division will become clearer as you read about additional details of each protocol and standard.

The two alternative TCP/IP models exist. Comparing the two, the upper layers are identical. The lower layers differ in that the single network access layer in one model is split into two layers to match the division of physical transmission details from the other functions. Figure 2-10 shows the two models again, with emphasis on these distinctions.

Figure 2-10 *Network Access Versus Data Link and Physical Layers*



Data Encapsulation Terminology

As you can see from the explanations of how HTTP, TCP, IP, and Ethernet do their jobs, each layer adds its own header (and sometimes trailer) to the data supplied by the higher layer. The term *encapsulation* refers to the process of putting headers (and sometimes trailers) around some data.

Many of the examples in this chapter show the encapsulation process. For instance, web server Larry encapsulated the contents of the home page inside an HTTP header in Figure 2-4. The TCP layer encapsulated the HTTP headers and data inside a TCP header in Figure 2-5. IP encapsulated the TCP headers and the data inside an IP header in Figure 2-7. Finally, the Ethernet network access layer encapsulated the IP packets inside both a header and a trailer in Figure 2-9.

The process by which a TCP/IP host sends data can be viewed as a five-step process. The first four steps relate to the encapsulation performed by the four TCP/IP layers, and the last step is the actual physical transmission of the data by the host. In fact, if you use the five-layer TCP/IP model, one step corresponds to the role of each layer. The steps are summarized in the following list:

Step 1 **Create and encapsulate the application data with any required application layer headers.** For example, the HTTP OK message can be returned in an HTTP header, followed by part of the contents of a web page.

Step 2 **Encapsulate the data supplied by the application layer inside a transport layer header.** For end-user applications, a TCP or UDP header is typically used.

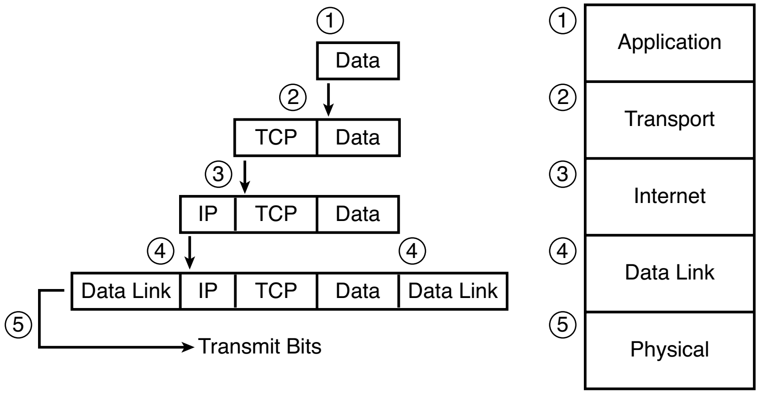
Step 3 **Encapsulate the data supplied by the transport layer inside an Internet layer (IP) header.** IP defines the IP addresses that uniquely identify each computer.

Step 4 **Encapsulate the data supplied by the Internet layer inside a data link layer header and trailer.** This is the only layer that uses both a header and a trailer.

Step 5 **Transmit the bits.** The physical layer encodes a signal onto the medium to transmit the frame.

The numbers in Figure 2-11 correspond to the five steps in this list, graphically showing the same concepts. Note that because the application layer often does not need to add a header, the figure does not show a specific application layer header.

Figure 2-11 *Five Steps of Data Encapsulation: TCP/IP*

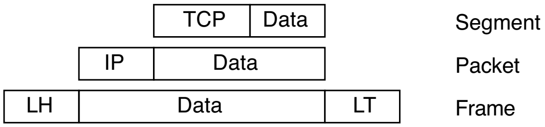


Names of TCP/IP Messages

Finally, take particular care to remember the terms *segment*, *packet*, and *frame*, and the meaning of each. Each term refers to the headers and possibly trailers defined by a particular layer, and the data encapsulated following that header. Each term, however, refers to a different layer: segment for the transport layer, packet for the Internet layer, and frame for the network access layer. Figure 2-12 shows each layer along with the associated term.

Figure 2-12 *Perspectives on Encapsulation and "Data"*

Key Topic



*The letters LH and LT stand for link header and link trailer, respectively, and refer to the data link layer header and trailer.

Figure 2-12 also shows the encapsulated data as simply "data." When focusing on the work done by a particular layer, the encapsulated data typically is unimportant. For example, an IP packet may indeed have a TCP header after the IP header, an HTTP header after the TCP header, and data for a web page after the HTTP header. However, when discussing IP, you probably just care about the IP header, so everything after the IP header is just called "data." So, when drawing IP packets, everything after the IP header is typically shown simply as "data."

OSI Networking Model

At one point in the history of the OSI model, many people thought OSI would win the battle of the networking models discussed earlier. If that had occurred, instead of running TCP/IP on every computer in the world, those computers would be running with OSI.

However, OSI did not win that battle. In fact, OSI no longer exists as a networking model that could be used instead of TCP/IP, although some of the original protocols referenced by the OSI model still exist.

So, why is OSI even in this book? Terminology. During those years in which many people thought the OSI model would become commonplace in the world of networking (mostly in the late 1980s and early 1990s), many vendors and protocol documents started using terminology from the OSI model. That terminology remains today. So, while you will never need to work with a computer that uses OSI, to understand modern networking terminology, you need to understand something about OSI.

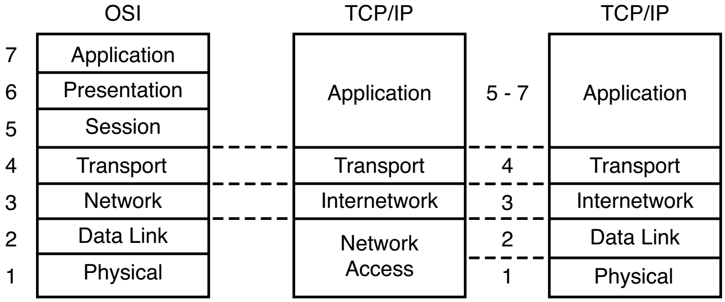
Comparing OSI and TCP/IP

The OSI model has many similarities to the TCP/IP model from a basic conceptual perspective. It has (seven) layers, and each layer defines a set of typical networking functions. As with TCP/IP, the OSI layers each refer to multiple protocols and standards that implement the functions specified by each layer. In other cases, just as for TCP/IP, the OSI committees did not create new protocols or standards, but instead referenced other protocols that were already defined. For instance, the IEEE defines Ethernet standards, so the OSI committees did not waste time specifying a new type of Ethernet; it simply referred to the IEEE Ethernet standards.

Today, the OSI model can be used as a standard of comparison to other networking models. Figure 2-13 compares the seven-layer OSI model with both the four-layer and five-layer TCP/IP models.

Key Topic

Figure 2-13 *OSI Model Compared to the Two TCP/IP Models*



Next, this section will examine two ways in which we still use OSI terminology today: to describe other protocols and to describe the encapsulation process. Along the way, the text will briefly examine each layer of the OSI model.

Describing Protocols by Referencing the OSI Layers

Even today, networking documents often describe TCP/IP protocols and standards by referencing OSI layers, both by layer number and layer name. For instance, a common description of a LAN switch is "layer 2 switch," with "layer 2" referring to OSI layer 2. Because OSI did have a well-defined set of functions associated with each of its seven layers, if you know those functions, you can understand what people mean when they refer to a product or function by its OSI layer.

For another example, TCP/IP's Internet layer, as implemented mainly by IP, equates most directly to the OSI *network* layer. So, most people say that IP is a *network layer protocol*, or a *Layer 3 protocol*, using OSI terminology and numbers for the layer. Of course, if you numbered the TCP/IP model, starting at the bottom, IP would be either Layer 2 or 3, depending on what version of the TCP/IP model you care to use. However, even though IP is a TCP/IP protocol, everyone uses the OSI model layer names and numbers when describing IP or any other protocol for that matter.

Although Figure 2-13 seems to imply that the OSI network layer and the TCP/IP Internet layer are at least similar, the figure does not point out why they are similar. To appreciate why the TCP/IP layers correspond to a particular OSI layer, you need to have a better understanding of the OSI layers. For example, the OSI network layer defines logical addressing and routing, as does the TCP/IP Internet layer. Although the details differ significantly, the TCP/IP Internet layer matches the overall goals and intent of the OSI network layer.

As another example, you may recall that the TCP/IP transport layer defines many functions, including error recovery. The OSI transport layer also defines these same functions as well, although with different details and different specific protocols. As a result, the networking industry refers to TCP as a Layer 4 protocol or a transport layer protocol, again based on the OSI layer number and name.

OSI Layers and Their Functions

Cisco requires that CCNAs demonstrate a basic understanding of the functions defined by each OSI layer, as well as remembering the names of the layers. It is also important that, for each device or protocol referenced throughout the book, you understand which layers of the OSI model most closely match the functions defined by that device or protocol.

Today, because most people happen to be much more familiar with TCP/IP functions than with OSI functions, one of the best ways to learn about the function of different OSI layers is to think about the functions in the TCP/IP model, and correlate those with the OSI model. If you use the five-layer TCP/IP model, the bottom four layers of OSI and TCP/IP map closely together. The only difference in these bottom four layers is the name of OSI Layer 3 (network) compared to TCP/IP (Internet). The upper three layers of the OSI reference model (application, presentation, and session—Layers 7, 6, and 5) define functions that all map to the TCP/IP application layer. Table 2-4 defines the functions of the seven layers.

Table 2-4 *OSI Reference Model Layer Definitions*

Layer	Functional Description
7	Layer 7 provides an interface between the communications software and any applications that need to communicate outside the computer on which the application resides. It also defines processes for user authentication.
6	This layer's main purpose is to define and negotiate data formats, such as ASCII text, EBCDIC text, binary, BCD, and JPEG. Encryption is also defined by OSI as a presentation layer service.
5	The session layer defines how to start, control, and end conversations (called sessions). This includes the control and management of multiple bidirectional messages so that the application can be notified if only some of a series of messages are completed. This allows the presentation layer to have a seamless view of an incoming stream of data.

Table 2-4 *OSI Reference Model Layer Definitions (Continued)*

Layer	Functional Description
4	Layer 4 protocols provide a large number of services, as described in Chapter 6, "Fundamentals of TCP/IP Transport, Applications, and Security." Although OSI Layers 5 through 7 focus on issues related to the application, Layer 4 focuses on issues related to data delivery to another computer (for instance, error recovery and flow control).
3	The network layer defines three main features: logical addressing, routing (forwarding), and path determination. Routing defines how devices (typically routers) forward packets to their final destination. Logical addressing defines how each device can have an address that can be used by the routing process. Path determination refers to the work done by routing protocols to learn all possible routes, and choose the best route.
2	The data link layer defines the rules that determine when a device can send data over a particular medium. Data link protocols also define the format of a header and trailer that allows devices attached to the medium to successfully send and receive data.
1	This layer typically refers to standards from other organizations. These standards deal with the physical characteristics of the transmission medium, including connectors, pins, use of pins, electrical currents, encoding, light modulation, and the rules for how to activate and deactivate the use of the physical medium.

Table 2-5 lists most of the devices and protocols covered in the CCNA exams and their comparable OSI layers. Note that many network devices must actually understand the protocols at multiple OSI layers, so the layer listed in Table 2-5 actually refers to the highest layer that the device normally thinks about when performing its core work. For example, routers need to think about Layer 3 concepts, but they must also support features at both Layers 1 and 2.

Table 2-5 *OSI Reference Model—Example Devices and Protocols*

Layer Name	Protocols and Specifications	Devices
Application, presentation, session (Layers 5–7)	Telnet, HTTP, FTP, SMTP, POP3, VoIP, SNMP	Firewall, intrusion detection systems, hosts
Transport (Layer 4)	TCP, UDP	Hosts, firewalls
Network (Layer 3)	IP	Router
Data link (Layer 2)	Ethernet (IEEE 802.3), HDLC, Frame Relay, PPP	LAN switch, wireless access point, cable modem, DSL modem
Physical (Layer 1)	RJ-45, EIA/TIA-232, V.35, Ethernet (IEEE 802.3)	LAN hub, LAN repeater, cables

Besides remembering the basics of the features of each OSI layer (as in Table 2-4), and some example protocols and devices at each layer (as in Table 2-5), you should also memorize the names of the layers. You can simply memorize them, but some people like to use a mnemonic phrase to make memorization easier. In the following three phrases, the first letter of each word is the same as the first letter of an OSI layer name, in the order specified in parentheses:

- All People Seem To Need Data Processing (Layers 7 to 1)
- Please Do Not Take Sausage Pizzas Away (Layers 1 to 7)
- Pew! Dead Ninja Turtles Smell Particularly Awful (Layers 1 to 7)

OSI Layering Concepts and Benefits

While networking models use layers to help humans categorize and understand the many functions in a network, networking models use layers for many reasons. For example, consider another postal service analogy. A person writing a letter does not have to think about how the postal service will deliver a letter across the country. The postal worker in the middle of the country does not have to worry about the contents of the letter. Likewise, networking models that divide functions into different layers enables one software package or hardware device to implement functions from one layer, and assume that other software/hardware will perform the functions defined by the other layers.

The following list summarizes the benefits of layered protocol specifications:

Key Topic

- **Less complex:** Compared to not using a layered model, network models break the concepts into smaller parts.
- **Standard interfaces:** The standard interface definitions between each layer allow for multiple vendors to create products that fill a particular role, with all the benefits of open competition.
- **Easier to learn:** Humans can more easily discuss and learn about the many details of a protocol specification.
- **Easier to develop:** Reduced complexity allows easier program changes and faster product development.
- **Multivendor interoperability:** Creating products to meet the same networking standards means that computers and networking gear from multiple vendors can work in the same network.
- **Modular engineering:** One vendor can write software that implements higher layers—for example, a web browser—and another vendor can write software that implements the lower layers—for example, Microsoft's built-in TCP/IP software in its OSs.

OSI Encapsulation Terminology

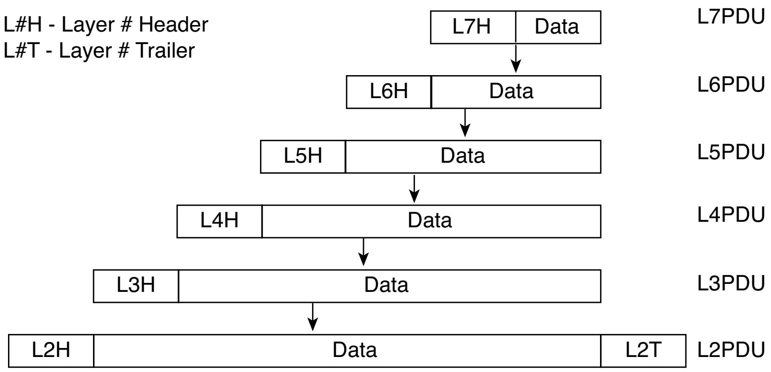
Like TCP/IP, each OSI layer asks for services from the next lower layer. To provide the services, each layer makes use of a header, and possibly a trailer. The lower layer encapsulates the higher layer's data behind a header. The final topic of this chapter explains some of the terminology and concepts related to OSI encapsulation.

The TCP/IP model uses terms such as *segment*, *packet*, and *frame* to refer to various layers and their respective encapsulated data (refer to Figure 2-11). OSI uses a more generic term: *protocol data unit* (*PDU*).

A PDU represents the bits that include the headers and trailers for that layer, as well as the encapsulated data. For instance, an IP packet, as shown in Figure 2-10, using OSI terminology, is a PDU. In fact, an IP packet is a *Layer 3 PDU* (abbreviated L3PDU) because IP is a Layer 3 protocol. So, rather than use the terms *segment*, *packet*, or *frame*, OSI simply refers to the "Layer x PDU" (LxPDU), with "x" referring to the number of the layer being discussed.

Figure 2-14 represents the typical encapsulation process, with the top of the figure showing the application data and application layer header and the bottom of the figure showing the L2PDU that is transmitted onto the physical link.

Figure 2-14 *OSI Encapsulation and Protocol Data Units*



Exam Preparation Tasks

Review All the Key Topics

Review the most important topics from inside the chapter, noted with the key topics icon in the outer margin of the page. Table 2-6 lists a reference of these key topics and the page number on which each is found.

Table 2-6 *Key Topics for Chapter 2*

Key Topic Elements	Description	Page Number
Table 2-3	Provides definitions of same-layer and adjacent-layer interaction	28
Figure 2-8	Shows the general concept of IP routing	32
Figure 2-9	Depicts the data-link services provided to IP for the purpose of delivering IP packets from host to host	33
Figure 2-12	Shows the meaning of the terms *segment*, *packet*, and *frame*	37
Figure 2-13	Compares the OSI and TCP/IP network models	38
List	Lists the benefits of using a layered networking model	41

Complete the Tables and Lists from Memory

Print a copy of Appendix L, "Memory Tables," (found on the DVD) or at least the section for this chapter, and complete the tables and lists from memory. Appendix M, "Memory Tables Answer Key," includes completed tables and lists to check your work.

Definitions of Key Terms

Define the following key terms from this chapter, and check your answers in the Glossary:

adjacent-layer interaction, decapsulation, encapsulation, frame, networking model, packet, protocol data unit (PDU), same-layer interaction, segment

OSI Reference

You should memorize the names of the layers of the OSI model. Table 2-7 summarizes the OSI functions at each layer, along with some sample protocols at each layer.

Table 2-7 *OSI Functional Summary*

Layer	**Functional Description**
Application (7)	Interfaces between network and application software. Also includes authentication services.
Presentation (6)	Defines the format and organization of data. Includes encryption.
Session (5)	Establishes and maintains end-to-end bidirectional flows between endpoints. Includes managing transaction flows.
Transport (4)	Provides a variety of services between two host computers, including connection establishment and termination, flow control, error recovery, and segmentation of large data blocks into smaller parts for transmission.
Network (3)	Logical addressing, routing, and path determination.
Data link (2)	Formats data into frames appropriate for transmission onto some physical medium. Defines rules for when the medium can be used. Defines means by which to recognize transmission errors.
Physical (1)	Defines the electrical, optical, cabling, connectors, and procedural details required for transmitting bits, represented as some form of energy passing over a physical medium.