The TCP/IP networking model

This chapter covers

- What networking models are and why we need them
- The OSI model
- The TCP/IP model and its layers
- How each layer plays a role in moving data across a network
- Data encapsulation and de-encapsulation

In the previous chapter, we looked at Ethernet; specifically, we looked at the types of physical connections defined by the Ethernet standard. Ethernet also defines rules for how devices can communicate over those connections. However, Ethernet alone isn't sufficient for two computers to communicate over a network (e.g., for a PC to retrieve a web page from a server over the internet). Communicating over a network is a complex process, and it requires a variety of protocols, each of which performs specific functions and, when brought together, enables network communications.

In this chapter, we will look at a couple of models that define the various functions required to enable computers to communicate over a network: the *Open Systems Interconnection* (OSI) model and the *TCP/IP* model (named after two key protocols of the model: Transmission Control Protocol and Internet Protocol). TCP/IP is the model currently used by modern networks all over the world.

Neither of these models is explicitly listed as a CCNA exam topic. However, the information in this chapter is fundamental networking knowledge. We will examine the functions of various network protocols throughout this book, so it's important to have a framework to understand it all. That's the role of these networking models - to provide a framework to organize the various functions that make a network work.

The purpose of this chapter is to provide a high-level overview of how data travels from source to destination across a network. In later chapters we will fill in the gaps regarding the exact mechanisms that make network communications possible, but first we need a framework.

4.1 Conceptual models of networking

Since the beginning of computer networking, there have been several attempts to create models which define the various functions necessary for computers to communicate with each other. Several of these models were vendor-proprietary, meaning they were created by a specific vendor (ie. IBM) to be used by their products. The vendor-proprietary approach, however, was not ideal; each vendor designed their own communication protocols, and enabling communication between different vendors' products was no simple task.

Definition

A *protocol* is a set of rules defining how data should be communicated between devices in a network. Protocols can be thought of as the languages computers use to communicate; two computers using different networking protocols are like two humans speaking different languages - they won't be able to communicate.

These days we all enjoy the benefits of the alternative approach: *vendor-neutral*. In a vendor-neutral model, with vendor-neutral protocols that can be used by devices of all kinds, we don't have to worry about whether an Apple MacBook will be able to access a website hosted on a Linux web server, or whether a PC running windows will be able to send an email that can be read on a smartphone running Android.

Networking models are frameworks which define the various functions needed to allow data to travel from source to destination over a network. These functions are typically divided into *layers*, each layer describing a certain role required to enable network communications. Then, protocols can be designed to fill those roles.

This allows for a modular design: at each layer of the model there are several protocols which can fill the necessary roles of the layer. For example, in the previous chapter we looked at some aspects of Ethernet (IEEE 802.3) and also briefly mentioned wireless LANs as defined by IEEE 802.11 (best known as Wi-Fi). Both of those protocols serve the same purpose; they define how data should be sent over a particular physical medium (UTP/fiber cables for Ethernet, radio waves for Wi-Fi). An email application on a computer doesn't need to care about whether a message will be sent over the network via a wired Ethernet connection or a wireless Wi-Fi connection; as long as the email application performs its role, it can expect the other layers to perform their roles as well.

There are two networking models which the modern network engineer should be familiar with: OSI and TCP/IP. Although the TCP/IP Model is the model used in modern networks, the OSI Model has also had a large influence on how we think and talk about networks, and several of the protocols defined in the OSI Model are still in use in modern networks, so the OSI Model should be considered core knowledge for anyone involved in networking.

4.2 The OSI reference model

The *Open Systems Interconnection Reference Model* is a conceptual model of networking developed by the *International Organization for Standardization* (ISO). In this book I will refer to it as the OSI Model.

International Organization for Standardization

The ISO publishes standards related to various aspects of technology. Looking at the name, you may wonder why it's abbreviated as "ISO" and not "IOS". They decided upon the abbreviation "ISO" to have one shared abbreviation regardless of language. Rather than being an acronym for International Organization for Standardization, they state that ISO is derived from the Greek word isos, meaning "equal".

The OSI Model defines seven layers, each with its own functions that contribute to the process of communicating over a network. Table 4.1 lists the seven layers of the OSI Model.

Table 4.1 The seven layers of the OSI model (view table figure)

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

Because this chapter focuses on the TCP/IP Model, we won't cover the role of each of the seven layers listed in table 4.1. Although the number of layers is different, the OSI and TCP/IP Models are quite similar, so we will cover each layer's role and associated protocols in the following section on the TCP/IP Model.

4.3 The TCP/IP model

The TCP/IP Model was born out of research and development funded by the United States Department of Defense's *Defense Advanced Research Projects Agency* (DARPA). It was then called the ARPANET Reference Model, but has since evolved into the *Internet Protocol Suite*, which was defined in *Request for Comments* (RFC) 1122. RFCs are documents published by an organization called the *Internet Engineering Task Force* (IETF) to define standard protocols for the Internet. Some more common names for this model are the *TCP/IP Suite*, *TCP/IP Model*, or just *TCP/IP*. TCP and IP are two of the foundational protocols included in the model, so they are often used to refer to it.

RFCs and the IETF

The *IETF* is an organization that defines the standard protocols that make up the TCP/IP Model. *RFCs* are the documents published by the IETF which define these protocols. Many of these documents are informational or experimental, and sometimes humorous (for an example, check out RFC 1149 at https://datatracker.ietf.org/doc/html/rfc1149, which describes how to send network messages using birds). However, some RFCs go on to be recognized as *Internet Standards*; these are the RFCs that define the protocols that make up the TCP/IP Model. For example, TCP, IP, and other well-known protocols like HTTPS (which you'll see at the beginning of the URL I copied above) are Internet Standards.

The TCP/IP Model as defined in RFC 1122 has four layers, however network engineers typically reference a five-layer TCP/IP Model. The five-layer version of the model, as indicated by the thick border in table 4.2, is what we will be using in this book. Table 4.2 lists the layers of the TCP/IP Model, as well as their equivalent OSI Model layers and some example protocols that belong to each layer of the model.

Table 4.2 The TCP/IP model (view table figure)

OSI model	Four-layer TCP/IP model	Five-layer TCP/IP model	Example protocols
Application	Application	Application	HTTP
Presentation			HTTPS
Session			FTP SSH
Transport	Transport	Transport	TCP UDP
Network	Internet	Network	IPv4 IPv6
Data Link	Link	Data Link	Ethernet 802.11 (Wi-Fi)

Note

The five-layer TCP/IP Model that we will be referring to in this book can be thought of as a combination of the OSI Model and the four-layer TCP/IP Model; the bottom four layers are identical to the OSI Model, but the OSI Model's top three layers are combined into a single Application layer like in the four-layer TCP/IP Model.

The example protocols listed in table 4.2 are some of the protocols we will cover in this book; they are just a few of the protocols you should know for the CCNA exam. I included them in table 4.2 for reference, but we will cover how they actually function in later chapters of this book. In this chapter, we will focus on understanding the role of each layer of the TCP/IP Model.

4.3.1 The layers of the TCP/IP model

Figure 4.1 A web browser on PC1 uses a layer 7 protocol (HTTPS) to request a web page from the web server on SRV1. Layers 2, 3, and 4 work together to deliver the message to the appropriate application on SRV1. Layer 1 is the medium over which the communication occurs.

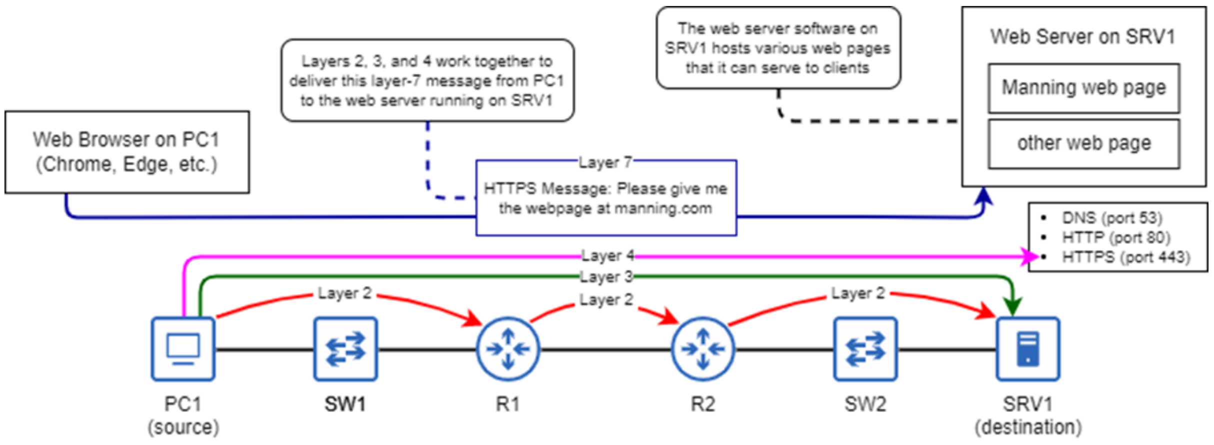
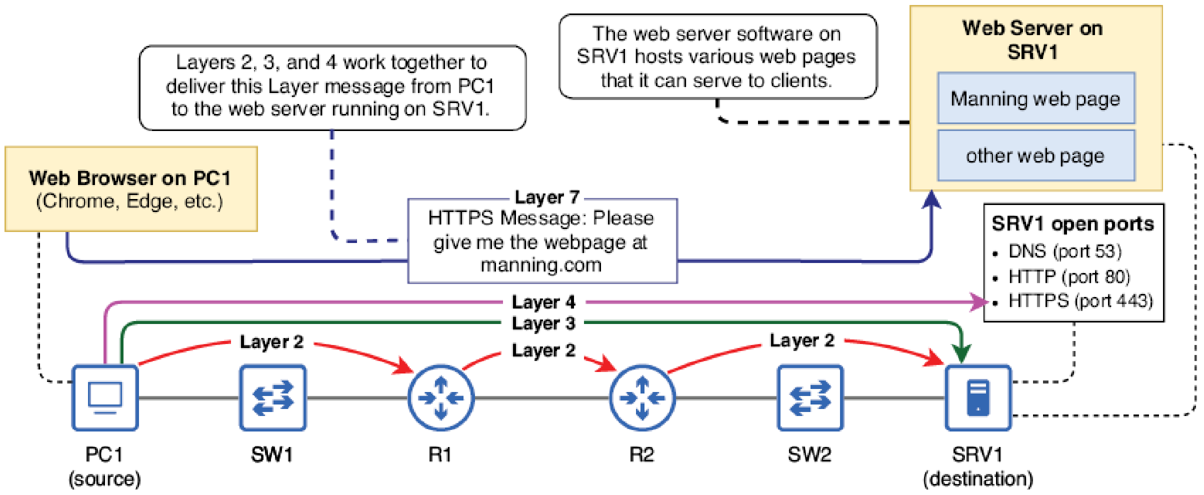

Web Browser on PC1 (Chrome, Edge, etc.)
Layers 2, 3, and 4 work together to deliver this layer-7 message from PC1 to the web server running on SRV1
The web server software on SRV1 hosts various web pages that it can serve to clients
Web Server on SRV1
Manning web page
other web page
Layer 7
HTTPS Message: Please give me the webpage at manning.com
DNS (port 53)
HTTP (port 80)
HTTPS (port 443)
Layer 4
Layer 3
Layer 2
PC1 (source)
SW1
R1
R2
SW2
SRV1 (destination)

Figure 4.1 A web browser on PC1 uses a Layer 7 protocol (HTTPS) to request a web page from the web server on SRV1. Layers 2, 3, and 4 work together to deliver the message to the appropriate application on SRV1. Layer 1 is the medium over which the communication occurs.


Layers 2, 3, and 4 work together to deliver this Layer message from PC1 to the web server running on SRV1.
The web server software on SRV1 hosts various web pages that it can serve to clients.
Web Server on SRV1
Manning web page
other web page
Web Browser on PC1 (Chrome, Edge, etc.)
Layer 7
HTTPS Message: Please give me the webpage at manning.com
SRV1 open ports
DNS (port 53)
HTTP (port 80)
HTTPS (port 443)
Layer 4
Layer 3
Layer 2
PC1 (source)
SW1
R1
R2
SW2
SRV1 (destination)

physical specifications, such as cables and radio waves

- communication between intermediate nodes in the path to the destination
- end-to-end communication from the original source node to the final destination node
- addressing messages to a specific application on the destination node
- how an application should interface with the network
- Now let's examine each layer of the TCP/IP Model one by one to see how they enable network communications. The goal of this chapter is to provide a framework which we can build upon with details of how the different protocols of each layer fulfill their roles in later chapters.

Layer 1: The Physical Layer

This is what we covered in chapter 3; IEEE 802.3 (Ethernet) and IEEE 802.11 (Wi-Fi) both define specifications at the Physical layer. For example, Ethernet defines connector and cable types, how data should be encoded into electrical (or light) signals, and countless other minutiae about how to communicate over UTP and fiber-optic cables. Likewise, Wi-Fi defines what radio frequencies should be used for wireless LAN communication, how radio waves should be modulated to encode data, etc.

To summarize the Physical layer of the TCP/IP Model: it defines the physical requirements to enable a series of bits to travel from one node to another over a physical medium.

Layer 2: The Data Link Layer

Figure 4.2 demonstrates the concept of network hops. PC1 sends a message to SRV1, perhaps a request to access a file hosted on the server. For PC1's message to reach SRV1, it must make three hops through the network: one from PC1 to R1, one from R1 to R2, and one from R2 to SRV1. It is the Data Link layer's job to forward the message from one hop to the next until it reaches the destination host: SRV1. Notice that a message traveling through a switch does not count as a hop. We will examine why this is when we look at *Ethernet LAN switching* in chapter 6.

Figure 4.2 TCP/IP Layer 2. A message sent from PC1 to SRV1 takes three hops through the network: from PC1 to R1, from R1 to R2, and from R2 to SRV1. At each hop, the message is addressed to the next hop's MAC address. A message traveling through a switch does not count as a hop.

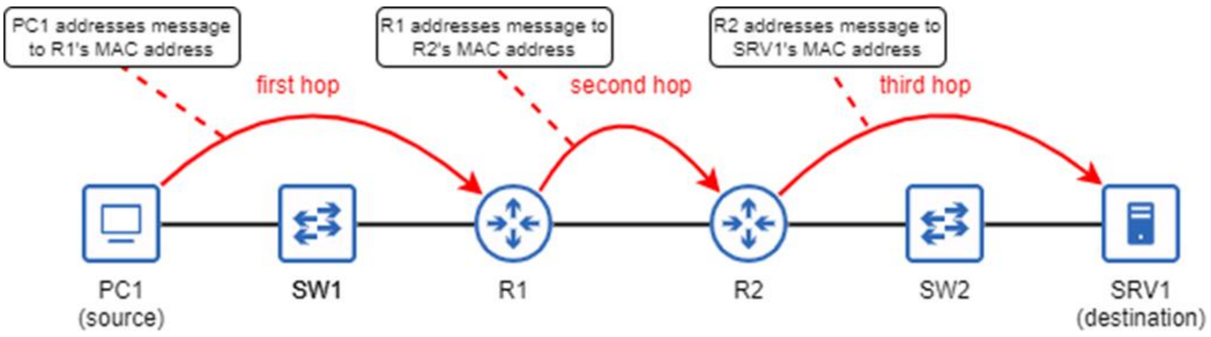
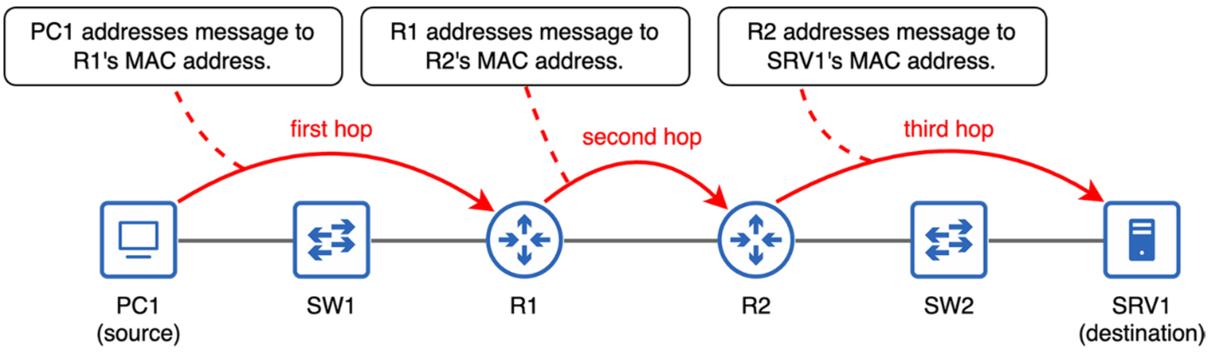

PC1 addresses message to R1's MAC address
R1 addresses message to R2's MAC address
R2 addresses message to SRV1's MAC address
first hop
second hop
third hop
PC1 (source)
SW1
R1
R2
SW2
SRV1 (destination)

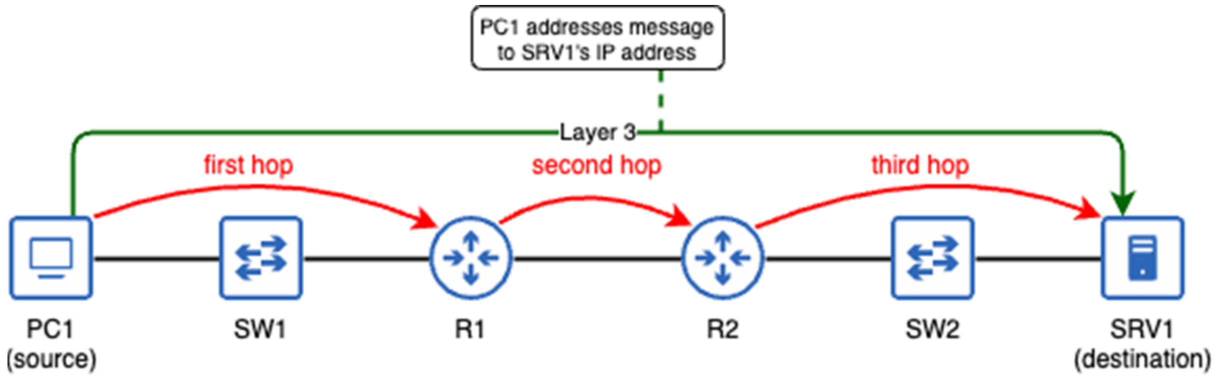
Figure 4.2 TCP/IP Layer 2. A message sent from PC1 to SRV1 takes three hops through the network: from PC1 to R1, from R1 to R2, and from R2 to SRV1. At each hop, the message is addressed to the next hop's MAC address. A message traveling through a switch does not count as a hop.


PC1 addresses message to R1's MAC address.
R1 addresses message to R2's MAC address.
R2 addresses message to SRV1's MAC address.
first hop
second hop
third hop
PC1 (source)
SW1
R1
R2
SW2
SRV1 (destination)

Layer 3: The Network Layer

The type of address used at the Network layer is the Internet Protocol (IP) address. Chances are you've heard of IP addresses before, although you might be unsure about how they work. We will cover IP addresses in chapter 7 of this book. Figure 4.2 shows how PC1 addresses a message to SRV1 by addressing it to SRV1's IP address. The destination IP address of the message remains the same throughout the journey, whereas the destination MAC address is different at each hop.

Figure 4.3 TCP/IP Layer 3. PC1 addresses a message to SRV1's IP address. Layer 3 is responsible for the end-to-end delivery of the message, whereas Layer 2 is responsible for the hop-to-hop delivery. The destination MAC address of the message changes at each hop, but the destination IP address remains the same throughout the journey.



IPv4 and IPv6

IPv4 address: 203.0.113.255

- JFo4 address: 203.0. 113. 255
- JLo6 address: 200 1:uh8:1:1:2lo3:1:32c:sl01

Understanding how layers 2 and 3 work together to deliver a message to its destination is a fundamental concept you must understand for the CCNA exam. In this chapter I just want to provide a high-level overview of the concepts; we will review these concepts and dig deeper in later chapters. At this point, it is enough to know the following points:

Layer 2 uses MAC addresses to provide hop-to-hop delivery of messages.

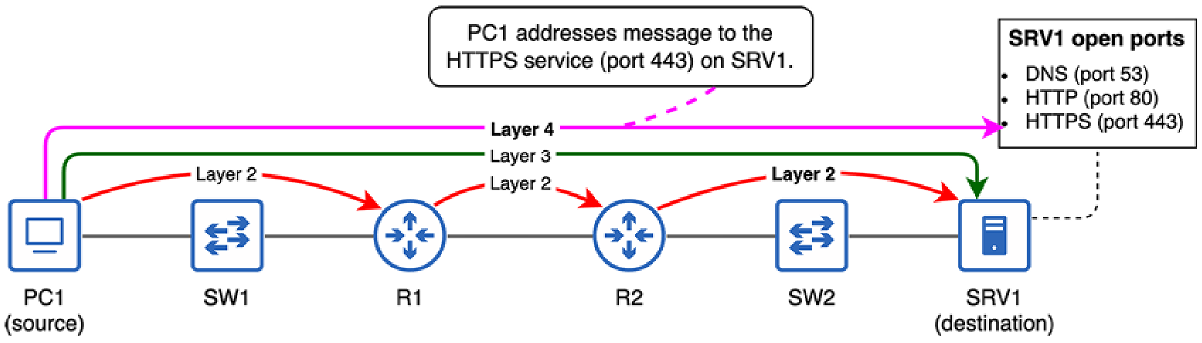
- Layer 3 uses IP addresses to provide end-to-end delivery of messages.
- Layers 2 and 3 work together to allow a message to travel through the network to its final destination.
- The destination IP address of a message remains the same throughout the journey, whereas the destination MAC address is different at each hop.

Layer 4: The Transport Layer

Like layers 2 and 3, layer 4 also uses its own addressing scheme: *port numbers*. By addressing a message to a particular port, you can send messages to a particular application process on the destination host. Computers run many different applications simultaneously, so this is a very important function. For example, on a PC can run an online game, a web browser with various tabs each accessing a different website, an antivirus application that communicates with an external server for updates, and countless other applications simultaneously. Port numbers allow the PC to ensure that data it receives from the network reaches the proper destination process.

Figure 4.3 demonstrates this concept. Layers 2 and 3 work together to deliver PC1's message to SRV1, and layer 4 delivers the message to the appropriate application process on SRV1. SRV1 is a server that provides a few services to clients in the network. It is a *name server* using *Domain Name System* (DNS) to convert website names to IP addresses for clients (that's what happens when you type *manning.com* into a web browser). It is also a web server using *Hypertext Transfer Protocol* (HTTP) and *Hypertext Transfer Protocol Secure* (HTTPS) to allow clients to access the websites it hosts. DNS, HTTP, and HTTPS are layer 7 (Application layer) protocols, and they each accept messages using a different layer 4 port number

Figure 4.4 Layers 2 and 3 work together to deliver PC1's message to SRV1. At Layer 4, PC1 addresses the message to port 443, which is used by the HTTPS protocol. Three ports are open on SRV1 (53, 80, 443), meaning it will accept messages addressed to any of those ports.



TCP and UDP

For example, TCP implements checks to ensure that each message reaches its destination, and is used by Application-layer protocols such as HTTP and HTTPS (used for accessing websites). UDP, on the other hand, takes a "send it and forget it" approach; it doesn't check to ensure that every message reaches the destination. UDP is used by *Voice over IP* (VoIP) protocols—used for phone calls—and live video streaming protocols, among others. We will cover TCP and UDP in chapter 20 of this book.

Layer 7: The Application Layer

Layer 7 protocols such as HTTPS are not applications themselves, rather they provide services for applications to enable them to communicate with applications on other computers over the network. Figure 4.1 showed the complete process that enables a web browser on PC1 to send a message to request a web page from the web server running on SRV1. The process that message goes through to reach SRV1 is as follows:

PC1's web browser uses HTTPS, a layer 7 protocol, to request the web page.

- At layer 4, PC1 addresses the message to port 443, which is used by the HTTPS protocol. This ensures that the message reaches the correct application on SRV1.
- At layer 3, PC1 addresses the message to the IP address of SRV1, and the destination IP address of the message remains the same as the message travels from PC1 across the network to SRV1.

- At layer 2, PC1 addresses the message to the next hop in the path to SRV1, which is R1. After receiving the message, R1 *forwards* it to the next hop (R2) by addressing the message to R2's MAC address. Finally, R2 forwards the message to the final destination (SRV1) by addressing the message to SRV1's MAC address. Unlike the destination IP address of the message, the destination MAC address is changed at each hop.

Definition

4.3.2 Data encapsulation and de-encapsulation

Layer 7 (Application): the interface between applications and the network

- Layer 4 (Transport): provides application-to-application delivery of messages
- Layer 3 (Network): provides end-to-end delivery of messages
- Layer 2 (Data Link): provides hop-to-hop delivery of messages
- Layer 1 (Physical): the physical medium over which communication happens

Data encapsulation

In the third step, the message is passed to layer 3, which adds its own header to that data. This header will be addressed to the IP address of the destination host. In the fourth step, the message will then be passed to layer 2, which adds both a header and a *trailer*.

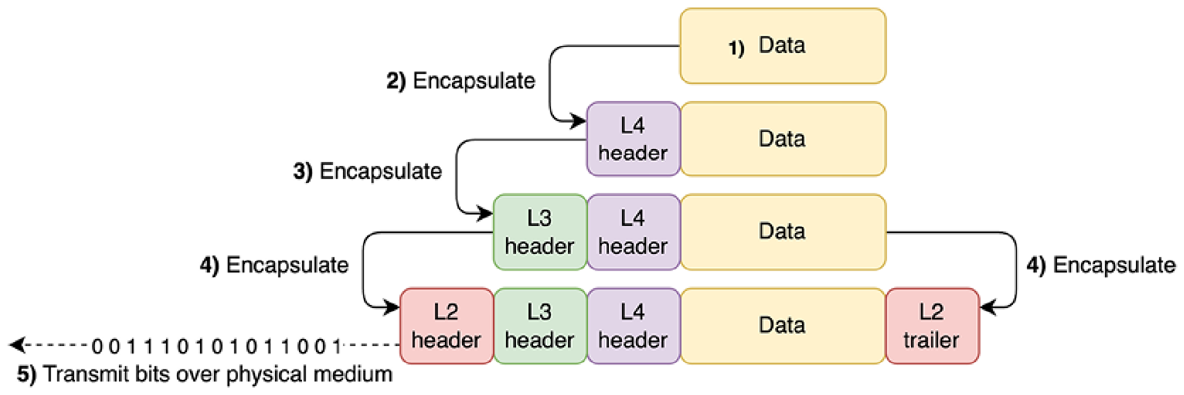
At layer 2, the message is addressed to the next-hop device. Finally, in the fifth step the host will transmit the bits over the physical medium, such as a UTP cable. The process of adding headers (and trailers) to data before sending it over a network is called *encapsulation*. To summarize that process:

The Application-layer protocol prepares data.

1. Layer 4 encapsulates the data with a header, addressed to a port number on the destination host.
2. Layer 3 encapsulates the data with a header, addressed to the IP address of the destination host.
3. Layer 2 encapsulates the data with a header, addressed to the MAC address of the next hop. It also encapsulates the data with a trailer, used to check for errors.
4. The host transmits the bits of data over the physical medium, for example encoded as electrical signals over a UTP cable.

Figure 4.5 demonstrates the five-step process of encapsulation and transmission.

Figure 4.5 The five-step process of encapsulating and transmitting data: (1) the Application Layer protocol prepares some data, (2) Layer 4 encapsulates the data with a header, (3) Layer 3 encapsulates the data with a header, (4) Layer 2 encapsulates the data with a header and trailer, and (5) the host transmits the bits over the physical medium (i.e., a UTP cable).



Data de-encapsulation

The destination host receives the message.

1. It inspects the layer 2 header and trailer, removes them, and passes the message to layer 3.
2. It inspects the layer 3 header, removes it, and passes the message to layer 4.
3. It inspects the layer 4 header, removes it, and sends the data to the appropriate application.
4. The application receives and processes the data.
5. **Figure 4.6 The five-step process of receiving and de-encapsulating data: 1) the destination host receives bits (the message), 2) the layer 2 header/trailer and inspected and removed, 3) the layer 3 header is inspected and removed, 4) the layer 4 header is inspected and removed, 5) the data is received and processed by the application**

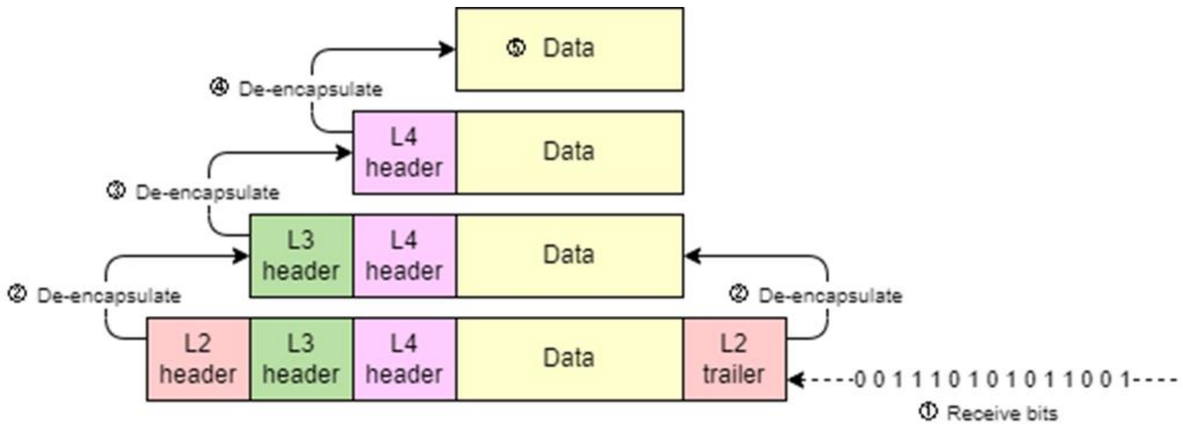
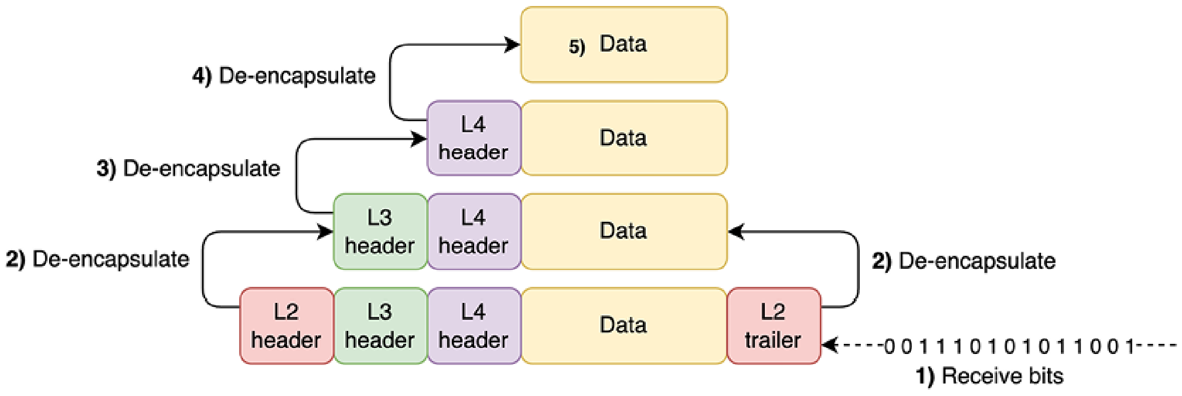


Figure 4.6 The five-step process of receiving and de-encapsulating data: (1) the destination host receives bits (the message), (2) the Layer 2 header/trailer is inspected and removed, (3) the Layer 3 header is inspected and removed, (4) the Layer 4 header is inspected and removed, and (5) the data is received and processed by the application.



Protocol data units

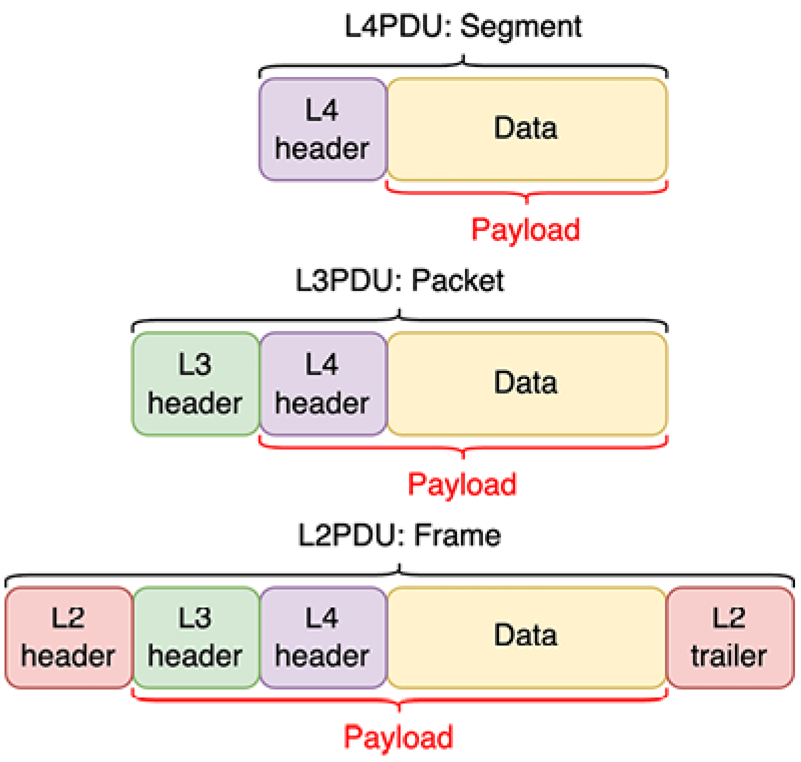
The combination of data and a layer 4 header is called a *segment*.

- The combination of a segment and a layer 3 header is called a *packet*.
- The combination of a packet and a layer 2 header/trailer is called a *frame*.
- We can also use a term from the OSI Model to describe the message at each stage: *protocol data unit* (PDU):

A segment is a layer 4 PDU (L4PDU).

- A packet is a layer 3 PDU (L3PDU).
- A frame is a layer 2 PDU (L2PDU).
- The contents of each PDU (everything encapsulated by that layer's header/trailer) is called the *payload*. So, a frame's payload is a packet, a packet's payload is a segment, and a segment's payload is the application data. Figure 4.7 illustrates the different PDUs and their payloads.

Figure 4.7 Application data encapsulated in a Layer 4 header is a segment (L4PDU); a segment encapsulated in a Layer 3 header is a packet (L3PDU); and a packet encapsulated in a Layer 2 header/trailer is a frame (L2PDU). The encapsulated contents of each PDU are that PDU's payload.



Adjacent-layer and same-layer interactions

Layer 4 provides a service to layer 7 by delivering data to the appropriate application on the destination host.

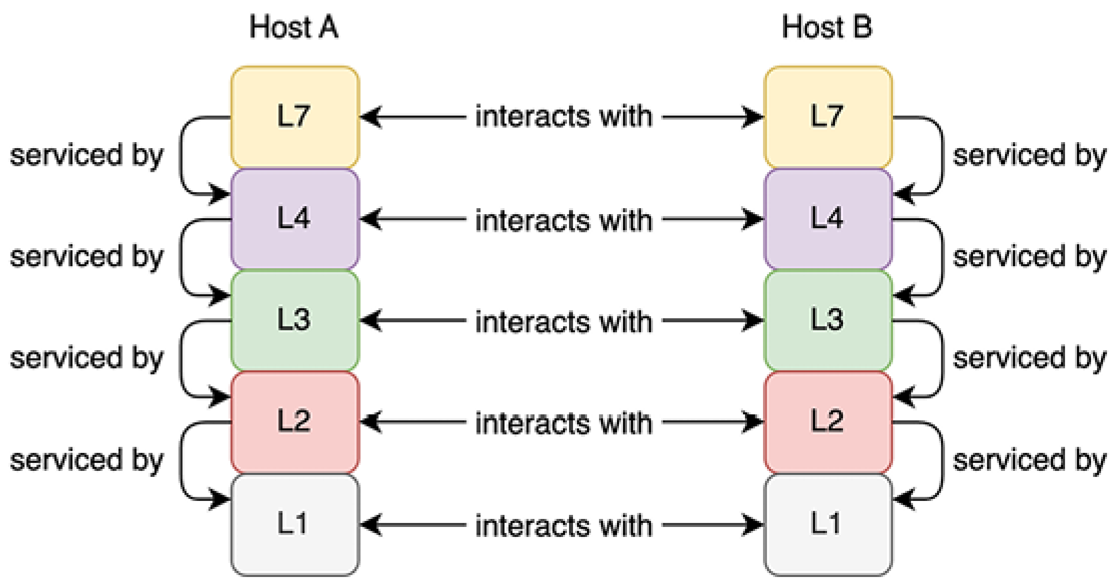
- Layer 3 provides a service to layer 4 by delivering segments to the correct destination host.
- Layer 2 provides to layer 3 by delivering packets to the next hop.
- Layer 1 provides a service to layer 2 by providing a physical medium for frames to travel over.
- There is also a related concept called *same-layer interaction*. This refers to the communications between the same layer on different computers. Same-layer interactions work like this:

Application data from one computer is sent to an application on another computer.

- When Data is encapsulated with a layer 4 header, the segment is addressed to layer 4 of the destination host, where the information in the header will be inspected.
- When a segment is encapsulated with a layer 3 header, the packet is addressed to layer 3 of the destination host, where the information in the header will be inspected.
- When a packet is encapsulated with a layer 2 header and trailer, the frame is addressed to layer 2 of the next hop, where the information in the header and trailer will be inspected.
- Bits sent out of a physical port of one device are received by a physical port of another device.

- Figure 4.8 illustrates these adjacent-layer interactions between different layers on the same computer (on Host A and on Host B), and same-layer interactions between different computers which are communicating with each other (between Host A and Host B).

Figure 4.8 Each layer on a host provides services for the layer above it; this is called adjacent-layer interaction. When two hosts communicate, each layer on one host communicates with the same layer on the other host; this is called same-layer interaction.



Summary

- Networking models provide frameworks to define the functions necessary to enable network communications.
- Networking models are divided into layers; each layer describes a necessary function for network communications and includes multiple protocols that can fulfill the layer’s role.
- The Open Systems Interconnection Reference (OSI) model is a networking model that influenced how we think and talk about networks but is not in use today.
- The OSI model has seven layers: (1) Physical, (2) Data Link, (3) Network, (4) Transport, (5) Session, (6) Presentation, and (7) Application.
- The Internet Protocol Suite (TCP/IP) model) is the networking model used in modern networks and is named after two of its key protocols: Transmission Control Protocol (TCP) and Internet Protocol (IP).
- The original TCP/IP model has four layers, but a more popular version has five: (1) Physical, (2) Data Link, (3) Network, (4) Transport, and (5) Application (called Layer 7, not Layer 5).
- Layer 1 (Physical) defines physical requirements for transmitting data, such as ports, connectors, and cables, and how data should be encoded into electrical/light signals.

- Layer 2 (Data Link) is responsible for hop-to-hop delivery of messages. A *hop* is the journey from one node in the network to the next in the path to the final destination.
- Layer 2 uses media access control (MAC) addresses to address messages to the next hop.
- Layer 3 (Network) is responsible for end-to-end delivery of messages, from the source host to the destination host.
- Layer 3 uses Internet Protocol (IP) addresses to address messages to the destination host.
- The destination MAC address of a message changes at each hop in the path to the destination, but the destination IP address remains the same.
- Layer 4 (Transport) is used to address messages to the appropriate application on the destination host.
- Layer 4's addressing scheme uses port numbers (not related to physical ports). The port number identifies the Layer 7 protocol being used.
- Layer 7 (Application) is the interface between applications and the network. Layer 7 protocols such as Hypertext Transfer Protocol Secure (HTTPS) are not applications themselves but provide services for applications to enable them to communicate over the network.
- A host *encapsulates* application data with a Layer 4 header, Layer 3 header, and Layer 2 header/trailer before being transmitted over the physical medium (cable or radio waves).
- After a message is received by a host, the host *de-encapsulates* it by inspecting and removing the Layer 2 header and trailer, inspecting and removing the Layer 3 header, inspecting and removing the Layer 4 header, and finally processing the data in the message.
- The contents encapsulated inside each *protocol data unit* (PDU) are its *payload.*
- The combination of data and a Layer 4 header is called a *segment* (L4PDU).
- The combination of a segment and a Layer 3 header is called a *packet* (L3PDU).
- The combination of a packet and a Layer 2 header/trailer is called a *frame* (L2PDU).
- Within a computer, each layer provides a service for the layer above it; this is called *adjacent-layer interaction*.
- Communication between the same layer on different computers is called *same-layer interaction*.