Joe Casad

Sams **Teach Yourself**

TCP/IP

in **24 Hours**

SIXTH EDITION

SAMS 800 East 96th Street, Indianapolis, Indiana 46240 USA

Sams Teach Yourself TCP/IP in 24 Hours, Sixth Edition

ISBN-13: 978-0-672-33789-5

ISBN-10: 0-672-33789-4

Library of Congress Control Number: 2016920083

1 17

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact

governmentsales@pearsoned.com

For questions about sales outside the U.S., please contact

intlcs@pearson.com

Editor-in-Chief
Greg Wiegand

Executive Editor
Laura Lewin

Development Editor
Michael Thurston

Managing Editor
Sandra Schroeder

Project Editor
Mandie Frank

Copy Editor
Abigail Bass

Indexer
Tim Wright

Proofreader
Larry Sulky

Technical Editors
Ronald McFarland
Jon Snader
Eric Spielman

Editorial Assistant
Olivia Basegio

Designer
Chuti Prasertsith

Compositor
codeMantra

Contents at a Glance

Table of Contents

APPENDIXES

About the Author

Joe Casad is an engineer, author, and editor who has written widely on computer networking and system administration. He has written or cowritten 12 books on computers and networking. He currently serves as editor-in-chief of *Linux Pro Magazine* and *ADMIN Magazine*. In a past life, he was the editor-in-chief of *C/C++ Users Journal* and the technical editor of *Sysadmin* Magazine.

Dedication

For Susan

Acknowledgments

Thanks to Laura Lewin, Olivia Basegio, Michael Thurston, Ronald McFarland, Jon Snader, Eric Spielman, Mandie Frank, Dhaya Karunanidhi, and Abby Manheim for their help with envisioning and creating this book. I also want to acknowledge the following individuals for their contributions to previous editions of *Sams Teach Yourself TCP/IP in 24 Hours*: Bob Willsey, Sudha Putnam, Walter Glenn, Art Hammond, Jane Brownlow, Jeff Koch, Mark Renfrow, Vicki Harding, Mark Cierzniak, Marc Charney, Jenny Watson, Betsy Harris, and Trina MacDonald. Thanks to Xander, Mattie, and Bridget for staying close in the storms and not wandering too far away in the sunshine. Thanks to my life partner Susan Rieger for venturing through canyons and over mountaintops with a guy who's still working on reading the map. And thanks with fond gratitude to the production department for bringing form and elegance to an inglorious collection of cryptic pencil sketches.

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

We welcome your comments. You can email or write to let us know what you did or didn't like about this book—as well as what we can do to make our books better.

Please note that we cannot help you with technical problems related to the topic of this book.

When you write, please be sure to include this book's title and author as well as your name and email address. We will carefully review your comments and share them with the author and editors who worked on the book.

Email: feedback@samspublishing.com

Mail: Sams Publishing
ATTN: Reader Feedback
800 East 96th Street
Indianapolis, IN 46240 USA

Reader Services

Register your copy of *Sams Teach Yourself TCP/IP in 24 Hours, Sixth Edition* at informit.com for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account*. Enter the product ISBN, 9780672337895, and click Submit. Once the process is complete, you will find any available bonus content under Registered Products.

*Be sure to check the box that you would like to hear from us in order to receive exclusive discounts on future editions of this product.

HOUR 2

How TCP/IP Works

What You'll Learn in This Hour:

- TCP/IP protocol system
- The OSI model
- Data packages
- How TCP/IP protocols interact

TCP/IP is a system (or suite) of protocols, and a protocol is a system of rules and procedures. For the most part, the hardware and software of the communicating computers carry out the rules of TCP/IP communications—the user does not have to get involved with the details. Still, a working knowledge of TCP/IP is essential if you want to navigate through the configuration and troubleshoot problems you'll face with TCP/IP networks.

This hour describes the TCP/IP protocol system and shows how the components of TCP/IP work together to send and receive data across the network.

At the completion of this hour, you will be able to

- Describe the layers of the TCP/IP protocol system and the purpose of each layer
- Describe the layers of the OSI model and explain how the OSI layers relate to TCP/IP
- Explain TCP/IP protocol headers and how data is enclosed with header information at each layer of the protocol stack
- Name the data package at each layer of the TCP/IP stack
- Discuss the TCP, UDP, and IP protocols and how they work together to provide TCP/IP functionality

The TCP/IP Protocol System

Before looking at the elements of TCP/IP, it is best to begin with a brief review of the responsibilities of a protocol system.

A protocol system such as TCP/IP must be responsible for the following tasks:

- Dividing messages into manageable chunks of data that will pass efficiently through the transmission medium.
- Interfacing with the network adapter hardware.
- Addressing: The sending computer must be capable of targeting data to a receiving computer. The receiving computer must be capable of recognizing a message that it is supposed to receive.
- Routing data to the subnet of the destination computer, even if the source subnet and the destination subnet are dissimilar physical networks.
- Performing error control, flow control, and acknowledgment: For reliable communication, the sending and receiving computers must be able to identify and correct faulty transmissions and control the flow of data.
- Accepting data from an application and passing it to the network.
- Receiving data from the network and passing it to an application.

To accomplish the preceding tasks, the creators of TCP/IP settled on a modular design. The TCP/IP protocol system is divided into separate components that theoretically function independently from one another. Each component is responsible for a piece of the communication process.

The advantage of this modular design is that it lets vendors easily adapt the protocol software to specific hardware and operating systems. For instance, the Network Access layer (as you learn in Hour 3, "The Network Access Layer") includes functions relating to the specification and design of the physical network. Because of TCP/IP's modular design, a vendor such as Microsoft does not have to build a completely different software package for TCP/IP on an optical-fiber network (as opposed to TCP/IP on an ordinary ethernet network). The upper layers are not affected by the different physical architecture; only the Network Access layer must change.

The TCP/IP protocol system is subdivided into layered components, each of which performs specific duties (see Figure 2.1). This model, or **stack**, comes from the early days of TCP/IP, and it is sometimes called the TCP/IP model. The official TCP/IP protocol layers and their functions are described in the following list. Compare the functions in the list with the responsibilities listed earlier in this section, and you'll see how the responsibilities of the protocol system are distributed among the layers.

BY THE WAY

Many Models

The four-layer model shown in Figure 2.1 is a common model for describing TCP/IP networking, but it isn't the only model. The ARPAnet model, for instance, as described in RFC 871, describes three layers: the Network Interface layer, the Host-to-Host layer, and the Process-Level/Applications layer. Other descriptions of TCP/IP call for a five-layer model, with Physical and Data Link layers in place of the Network Access layer (to match OSI). Still other models might exclude either the Network Access or the Application layer, which are less uniform and harder to define than the intermediate layers.

The names of the layers also vary. The ARPAnet layer names still appear in some discussions of TCP/IP, and the Internet layer is sometimes called the Internetwork layer or the Network layer.

This book uses the four-layer model, with names shown in Figure 2.1.

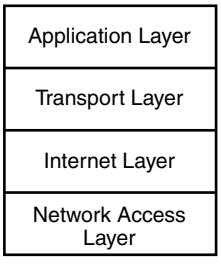

FIGURE 2.1
The TCP/IP model's protocol layers.

- **Network Access layer:** Provides an interface with the physical network. Formats the data for the transmission medium and addresses data for the subnet based on physical hardware addresses. Provides error control for data delivered on the physical network.
- **Internet layer:** Provides logical, hardware-independent addressing so that data can pass among subnets with different physical architectures. Provides routing to reduce traffic and support delivery across the internetwork. (The term **internetwork** refers to an interconnected, greater network of local area networks (LANs), such as what you find in a large company or on the Internet.) Relates physical addresses (used at the Network Access layer) to logical addresses.
- **Transport layer:** Provides flow-control, error-control, and acknowledgment services for the internetwork. Serves as an interface for network applications.
- **Application layer:** Provides applications for network troubleshooting, file transfer, remote control, and Internet activities. Also supports the network application programming interfaces (APIs) that enable programs written for a particular operating environment to access the network.

Later hours provide more detailed descriptions of the activities at each of these TCP/IP protocol layers.

When the TCP/IP protocol software prepares a piece of data for transmission across the network, each layer on the sending machine adds a layer of information to the data that is relevant to the corresponding layer on the receiving machine. For instance, the Internet layer of the computer sending the data adds a header with some information that is significant to the Internet layer of the computer receiving the message. This process is sometimes referred to as encapsulation. At the receiving end these headers are removed as the data is passed up the protocol stack.

BY THE WAY

Layers

The term *layer* is used throughout the computer industry for protocol component levels such as the ones shown in Figure 2.1. Header information is applied in layers to the data as it passes through the components of the protocol stack. (You'll learn more about this later in this hour.) When it comes to the components themselves, however, the term *layer* is somewhat metaphorical.

Diagrams such as Figure 2.1 are meant to show that the data passes across a series of interfaces. As long as the interfaces are maintained, the processes within one component are not affected by the processes in other components. If you turned Figure 2.1 sideways, it would look more like an assembly line, and this is also a useful analogy for the relationship of the protocol components. The data proceeds through a series of steps in the line and, as long as it arrives at each step as specified, the components can operate independently.

TCP/IP and the OSI Model

The networking industry has a standard seven-layer model for network protocol architecture called the Open Systems Interconnection (OSI) model. The OSI model represents an effort by the International Organization for Standardization (ISO), an international standards organization, to standardize the design of network protocol systems to promote interconnectivity and open access to protocol standards for software developers.

TCP/IP was already on the path of development when the OSI standard architecture appeared and, strictly speaking, TCP/IP does not conform to the OSI model. However, the two models did have similar goals, and enough interaction occurred among the designers of these standards that they emerged with a certain compatibility. The OSI model has been very influential in the growth and development of protocol implementations, and it is quite common to see the OSI terminology applied to TCP/IP.

Figure 2.2 shows the relationship between the four-layer TCP/IP standard and the seven-layer OSI model. Note that the OSI model divides the duties of the Application layer into three layers: Application, Presentation, and Session. OSI splits the activities of the Network Access layer into a Data Link layer and a Physical layer. This increased subdivision adds some complexity, but it also adds flexibility for developers by targeting the protocol layers to more specific services. In particular, the division at the lower level into the Data Link and Physical layers separates

the functions related to organizing communication from the functions related to accessing the communication medium. The three upper OSI layers offer a greater variety of alternatives for an application to interface with the protocol stack.

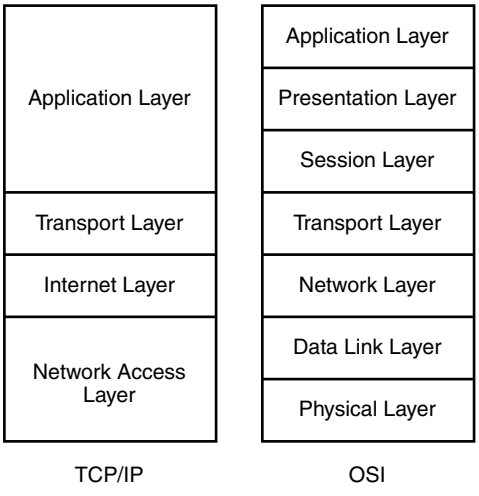

FIGURE 2.2
The seven-layer OSI model.

The seven layers of the OSI model are as follows:

- **Physical layer:** Converts the data into the stream of electrical or analog pulses that will actually cross the transmission medium and oversees the transmission of the data
- **Data Link layer:** Provides an interface with the network adapter; maintains logical links for the subnet
- **Network layer:** Supports logical addressing and routing
- **Transport layer:** Provides error control and flow control for the internetwork
- **Session layer:** Establishes sessions between communicating applications on the communicating computers
- **Presentation layer:** Translates data to a standard format; manages encryption and data compression
- **Application layer:** Provides a network interface for applications; supports network applications for file transfer, communications, and so forth

It is important to remember that the TCP/IP model and the OSI model are standards, not implementations. Real-world implementations of TCP/IP do not always map cleanly to the models shown in Figures 2.1 and 2.2, and the perfect correspondence depicted in Figure 2.2 is also a matter of some discussion within the industry.

Notice that the OSI and TCP/IP models are most similar at the important Transport and Internet (called Network in OSI) layers. These layers include the most identifiable and distinguishing components of the protocol system, and it is no coincidence that protocol systems are sometimes named for their Transport and Network layer protocols. As you will learn later in this book, the TCP/IP protocol suite is named for TCP, a Transport layer protocol, and IP, an Internet/Network layer protocol.

Data Packages

The important thing to remember about the TCP/IP protocol stack is that each layer plays a role in the overall communication process. Each layer invokes services that are necessary for that layer to perform its role. As an outgoing transmission passes down through the stack, each layer includes a bundle of relevant information called a **header** along with the actual data. The little data package containing the header and the data then becomes the data that is repackaged at the next lower level with the next lower layer's header. This process is shown in Figure 2.3. The reverse process occurs when data is received on the destination computer. As the data moves up through the stack, each layer unpacks the corresponding header and uses the information.

As the data moves down through the stack, the effect is a little like the nested Russian wooden dolls you might have seen; the innermost doll is enclosed in another doll, which is then enclosed in another doll, and so on. At the receiving end, the data packages are unpacked, one by one, as the data climbs back up the protocol stack. The Internet layer on the receiving machine uses the information in the Internet layer header. The Transport layer uses the information in the Transport layer header. At each layer, the package of data takes a form that provides the necessary information to the corresponding layer on the receiving machine. Because each layer is responsible for different functions, the form of the basic data package is very different at each layer.

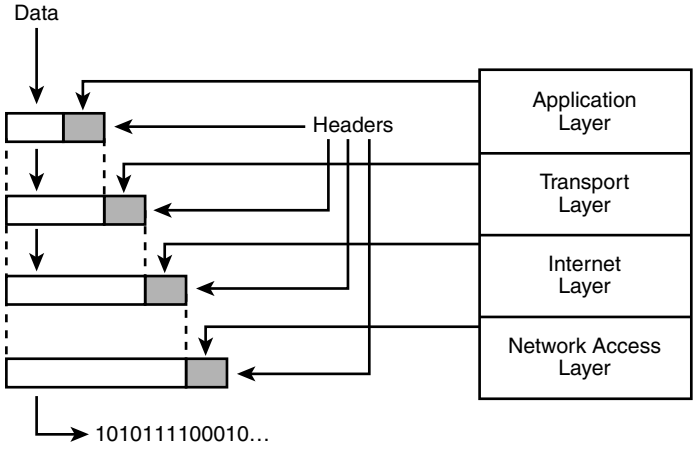

FIGURE 2.3
At each layer, the data is repackaged with that layer's header.

BY THE WAY

Transporting Dolls

The networking industry has as many analogies as it has acronyms, and the Russian doll analogy, like any of the others, illustrates a point, but must not be taken too far. It is worth noting that on a physical network such as ethernet, the data is typically broken into smaller units at the Network Access layer. A more accurate analogy would call for this lowest layer to break the concentric doll system into smaller pieces, encapsulate those pieces into tinier dolls, and then grind those tiny dolls into a pattern of 1s and 0s. The 1s and 0s are received, reconstituted into tiny dolls, and rebuilt into the concentric doll system. The complexity of this scenario causes many to eschew the otherwise-promising analogy of the dolls.

The data packet looks different at each layer, and at each layer it goes by a different name. The names for the data packages created at each layer are as follows:

- The data package created at the Application layer is called a **message**.
- The data package created at the Transport layer, which encapsulates the Application layer message, is called **a segment** if it comes from the Transport layer's TCP protocol. If the data package comes from the Transport layer's **User Datagram Protocol (UDP)** protocol, it is called a **datagram**.
- The data package at the Internet layer, which encapsulates the Transport layer segment, is called a datagram.
- The data package at the Network Access layer, which encapsulates and may subdivide the datagram, is called a **frame**. This frame is then turned into a bitstream at the lowest sub-layer of the Network Access layer.

To be honest, people don't always use these different protocol package names anymore; the word "packet" has become a popular (if imprecise) shorthand for describing a data package at any protocol level, but it is still worthwhile to consider that the different protocol packages have different names because they are actually quite different. Each layer has a different purpose, and each header contains different information. You learn more about the data packages for each layer in later hours.

A Quick Look at TCP/IP Networking

The practice of describing protocol systems in terms of their layers is widespread and nearly universal. The layering system does provide insights into the protocol system, and it's impossible to describe TCP/IP without first introducing its layered architecture. However, focusing solely on protocol layers also creates some limitations.

First, talking about protocol layers rather than protocols introduces additional abstraction to a subject that is already excruciatingly abstract. Second, itemizing the various protocols as subheads within the greater topic of a protocol layer can give the false impression that all protocols are of equal importance. In fact, though every protocol has a role to play, most of the functionality of the TCP/IP suite can be described in terms of only a few of its most important protocols. It is sometimes useful to view these important protocols in the foreground, against the backdrop of the layering system described earlier in this hour.

Figure 2.4 describes the basic TCP/IP protocol networking system. Of course, there are additional protocols and services in the complete package, but Figure 2.4 shows most of what is going on.

The basic scenario is as follows:

1. Data passes from a protocol, network service, or application programming interface (API) operating at the Application layer through a TCP or UDP port to either of the two Transport layer protocols (TCP or UDP). Programs can access the network through either TCP or UDP, depending on the program's requirements:
    - **TCP** is a connection-oriented protocol. As you learn in Hour 6, "The Transport Layer," connection-oriented protocols provide more sophisticated flow control and error control than connectionless protocols. TCP goes to great effort to guarantee the delivery of the data. TCP is more reliable than UDP, but the additional error checking and flow control mean that TCP is slower than UDP.
    - **UDP** is a connectionless protocol. It is faster than TCP, but it is not as reliable. UDP offloads more of the error control responsibilities to the application.
2. The data segment passes to the Internet level, where the **IP** protocol provides logical-addressing information and encloses the data into a datagram.
3. The IP datagram enters the Network Access layer, where it passes to software components designed to interface with the physical network. The Network Access layer creates one or more data frames designed for entry onto the physical network. In the case of a LAN system such as ethernet, the frame may contain physical address information obtained from lookup tables maintained using the Internet layer **ARP** protocol. (ARP, Address Resolution Protocol, translates IP addresses to physical addresses.)
4. The data frame is converted to a stream of bits that is transmitted over the network medium.

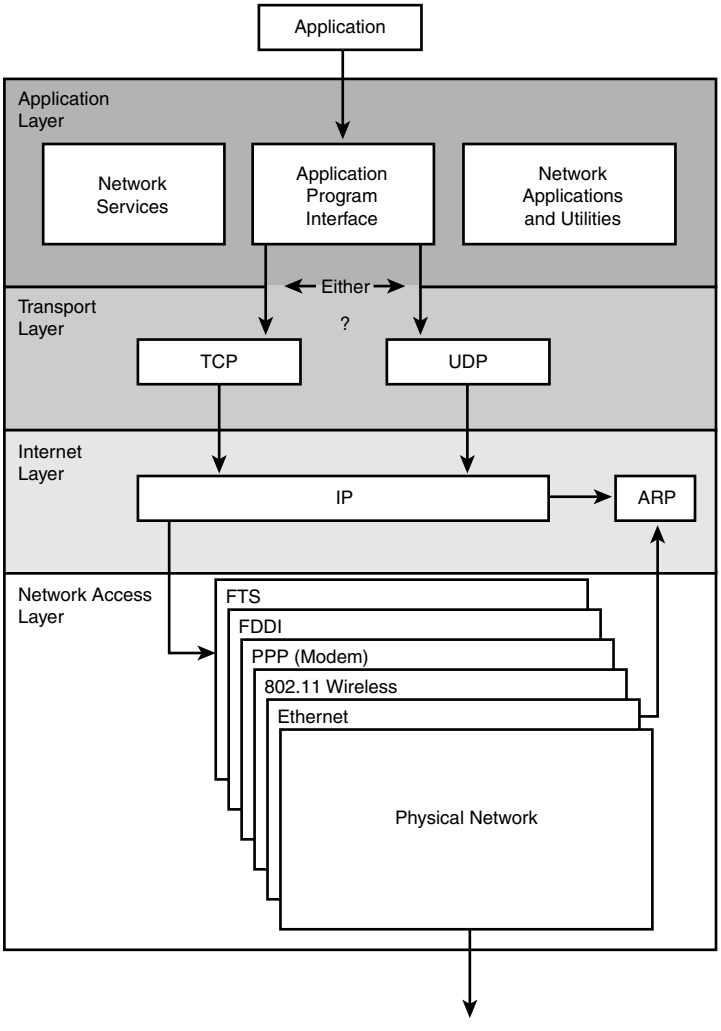

FIGURE 2.4
A quick look at the basic TCP/IP networking system.

Of course, there are endless details describing how each protocol goes about fulfilling its assigned tasks. For instance, how does TCP provide flow control, how does ARP map physical addresses to IP addresses, and how does IP know where to send a datagram addressed to a different subnet? These questions are explored later in this book.

Summary

In this hour, you learned about the layers of the TCP/IP protocol stack and how those layers interrelate. You also learned how the classic TCP/IP model relates to the seven-layer OSI networking model. At each layer in the protocol stack, data is packaged into the form that is most useful to the corresponding layer on the receiving end. This hour discussed the process of encapsulating header information at each protocol layer and outlined the different terms used at each layer to describe the data package. Finally, you got a quick look at how the TCP/IP protocol system operates from the viewpoint of some of its most important protocols: TCP, UDP, IP, and ARP.

Q&A

Q. What are the principal advantages of TCP/IP's modular design?

A. Because of TCP/IP's modular design, the TCP/IP protocol stack can adapt easily to specific hardware and operating environments. One layer can change without affecting the rest of the stack. Breaking the networking software into specific, well designed components also makes it easier to write programs that interact with the protocol system.

Q. What functions are provided at the Network Access layer?

A. The Network Access layer provides services related to the specific physical network. These services include preparing, transmitting, and receiving the frame over a particular transmission medium, such as an ethernet cable.

Q. Which OSI layer corresponds to the TCP/IP Internet layer?

A. TCP/IP's Internet layer corresponds to the OSI Network layer.

Q. Why is header information enclosed at each layer of the TCP/IP protocol stack?

A. Because each protocol layer on the receiving machine needs different information to process the incoming data, each layer on the sending machine encloses header information.

Workshop

The following workshop is composed of a series of quiz questions and practical exercises. The quiz questions are designed to test your overall understanding of the current material. The practical exercises are intended to afford you the opportunity to apply the concepts discussed during the current hour, as well as build upon the knowledge acquired in previous hours of study. Please take time to complete the quiz questions and exercises before continuing. Refer to Appendix A, "Answers to Quizzes and Exercises," for answers.

Quiz

1. What two OSI layers map into the TCP/IP Network Access layer?
2. What TCP/IP layer is responsible for routing data from one network segment to another?
3. What are the advantages and disadvantages of UDP as compared to TCP?
4. What does it mean to say that a layer encapsulates data?

Exercises

1. List the functions performed by each layer in the TCP/IP stack.
2. List the layer(s) that deal with datagrams.
3. Explain how TCP/IP would have to change to use a newly invented type of network hardware.
4. Explain what it means to say that TCP is a reliable protocol.

Key Terms

Review the following list of key terms:

- **Address Resolution Protocol (ARP):** A protocol that resolves logical IP addresses to physical addresses.
- **Application layer:** The layer of the TCP/IP stack that supports network applications and provides an interface to the local operating environment.
- **Datagram:** The data package passed between the Internet layer and the Network Access layer, or a data package passed between UDP at the Transport layer and the Internet layer.
- **Frame:** The data package created at the Network Access layer.
- **Header:** A bundle of protocol information attached to the data at each layer of the protocol stack.
- **Internet layer:** The layer of the TCP/IP stack that provides logical addressing and routing.
- **IP (Internet Protocol):** The Internet layer protocol that provides logical addressing and routing capabilities.
- **Message:** In TCP/IP networking, a message is the data package passed between the Application layer and the Transport layer. The term is also used generically to describe a message from one entity to another on the network. The term doesn't always refer to an Application layer data package.

- **Network Access layer:** The layer of the TCP/IP stack that provides an interface with the physical network.
- **Segment:** The data package passed between TCP at the Transport layer and the Internet layer.
- **TCP (Transmission Control Protocol):** A reliable, connection-oriented protocol of the Transport layer.
- **Transport layer:** The layer of the TCP/IP stack that provides error control and acknowledgment and serves as an interface for network applications.
- **UDP (User Datagram Protocol):** An unreliable, connectionless protocol of the Transport layer.

Index

Symbols

Numbers

A

B

c

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z