CISCO™

Course Booklet

CCNA Exploration
Network Fundamentals

Version 4.0

Cisco | Networking Academy
Mind Wide Open

CCNA Exploration Course Booklet
Network Fundamentals Version 4.0
Cisco Networking Academy

Warning and Disclaimer

This book is designed to provide information about Networking. 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 is provided on an "as is" basis. The authors, Cisco Press, and Cisco Systems, Inc. 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 or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Cisco Systems, Inc.

This book is part of the Cisco Networking Academy® series from Cisco Press. The products in this series support and complement the Cisco Networking Academy curriculum. If you are using this book outside the Networking Academy, then you are not preparing with a Cisco trained and authorized Networking Academy provider.

For more information on the Cisco Networking Academy or to locate a Networking Academy, Please visit www.cisco.com/edu.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc., 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.

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Welcome

Welcome to the CCNA Exploration Network Fundamentals course. The goal of this course is to introduce you to fundamental networking concepts and technologies. These online course materials will assist you in developing the skills necessary to plan and implement small networks across a range of applications. The specific skills covered in each chapter are described at the start of each chapter.

More than just information

This computer-based learning environment is an important part of the overall course experience for students and instructors in the Networking Academy. These online course materials are designed to be used along with several other instructional tools and activities. These include:

- Class presentation, discussion, and practice with your instructor
- Hands-on labs that use networking equipment within the Networking Academy classroom
- Online scored assessments and grade book
- Packet Tracer 4.1 simulation tool
- Additional software for classroom activities

A global community

When you participate in the Networking Academy, you are joining a global community linked by common goals and technologies. Schools, colleges, universities and other entities in over 160 countries participate in the program. You can see an interactive network map of the global Networking Academy community at http://www.academynetspace.com.

The material in this course encompasses a broad range of technologies that facilitate how people work, live, play, and learn by communicating with voice, video, and other data. Networking and the Internet affect people differently in different parts of the world. Although we have worked with instructors from around the world to create these materials, it is important that you work with your instructor and fellow students to make the material in this course applicable to your local situation.

Keep in Touch

These online instructional materials, as well as the rest of the course tools, are part of the larger Networking Academy. The portal for the program is located at http://cisco.netacad.net. There you will obtain access to the other tools in the program such as the assessment server and student grade book), as well as informational updates and other relevant links.

Mind Wide Open®

An important goal in education is to enrich you, the student, by expanding what you know and can do. It is important to realize, however, that the instructional materials and the instructor can only

facilitate the process. You must make the commitment yourself to learn new skills. Below are a few suggestions to help you learn and grow.

1. Take notes. Professionals in the networking field often keep Engineering Journals in which they write down the things they observe and learn. Taking notes is an important way to help your understanding grow over time.

2. Think about it. The course provides information both to change what you know and what you can do. As you go through the course, ask yourself what makes sense and what doesn't. Stop and ask questions when you are confused. Try to find out more about topics that interest you. If you are not sure why something is being taught, consider asking your instructor or a friend. Think about how the different parts of the course fit together.

3. Practice. Learning new skills requires practice. We believe this is so important to e-learning that we have a special name for it. We call it e-doing. It is very important that you complete the activities in the online instructional materials and that you also complete the hands-on labs and Packet Tracer® activities.

4. Practice again. Have you ever thought that you knew how to do something and then, when it was time to show it on a test or at work, you discovered that you really hadn't mastered it? Just like learning any new skill like a sport, game, or language, learning a professional skill requires patience and repeated practice before you can say you have truly learned it. The online instructional materials in this course provide opportunities for repeated practice for many skills. Take full advantage of them. You can also work with your instructor to extend Packet Tracer, and other tools, for additional practice as needed.

5. Teach it. Teaching a friend or colleague is often a good way to reinforce your own learning. To teach well, you will have to work through details that you may have overlooked on your first reading. Conversations about the course material with fellow students, colleagues, and the instructor can help solidify your understanding of networking concepts.

6. Make changes as you go. The course is designed to provide feedback through interactive activities and quizzes, the online assessment system, and through interactions with your instructor. You can use this feedback to better understand where your strengths and weaknesses are. If there is an area that you are having trouble with, focus on studying or practicing more in that area. Seek additional feedback from your instructor and other students.

Explore the world of networking

This version of the course includes a special tool called Packet Tracer 4.1®. Packet Tracer is a networking learning tool that supports a wide range of physical and logical simulations. It also provides visualization tools to help you to understand the internal workings of a network.

The Packet Tracer activities included in the course consist of network simulations, games, activities, and challenges that provide a broad range of learning experiences.

Create your own worlds

You can also use Packet Tracer to create your own experiments and networking scenarios. We hope that, over time, you consider using Packet Tracer – not only for experiencing the activities included in the course, but also to become an author, explorer, and experimenter.

The online course materials have embedded Packet Tracer activities that will launch on computers running Windows® operating systems, if Packet Tracer is installed. This integration may also work on other operating systems using Windows emulation.

Course Overview

As the course title states, the focus of this course is on learning the fundamentals of networking. In this course, you will learn both the practical and conceptual skills that build the foundation for understanding basic networking. First, you will examine human versus network communication and see the parallels between them. Next, you will be introduced to the two major models used to plan and implement networks: OSI and TCP/IP. You will gain an understanding of the "layered" approach to networks and examine the OSI and TCP/IP layers in detail to understand their functions and services. You will become familiar with the various network devices, network addressing schemes and, finally, the types of media used to carry data across the network.

In this course, you will gain experience using networking utilities and tools, such as Packet Tracer and Wireshark®, to explore networking protocols and concepts. These tools will help you to develop an understanding of how data flows in a network. A special "model Internet" is also used to provide a test environment where a range of network services and data can be observed and analyzed.

Chapter 1 - Chapter 1 presents the basics of communication and how networks have changed our lives. You will be introduced to the concepts of networks, data, local area networks (LANs), wide area networks (WANs), quality of service (QoS), security issues, network collaboration services, and Packet Tracer activities. In the labs, you will learn how to set up a wiki and establish an instant messaging session.

Chapter 2 - Chapter 2 focuses on how networks are modeled and used. You will be introduced to the OSI and TCP/IP models and to the process of data encapsulation. You will learn about the network tool Wireshark®, which is used for analyzing network traffic, and will explore the differences between a real network and a simulated network. In the lab, you will build your first network – a small peer-to-peer network.

Chapter 3 - Using a top-down approach to teaching networking, Chapter 3 introduces you to the top network model layer, the Application layer. In this context, you will explore the interaction of protocols, services, and applications, with a focus on HTTP, DNS, DHCP, SMTP/POP, Telnet and FTP. In the labs, you will practice installing a web server/client and use Wireshark® to analyze network traffic. The Packet Tracer activities let you explore how protocols operate at the Application layer.

Chapter 4 – Chapter 4 introduces the Transport layer and focuses on how the TCP and UDP protocols apply to the common applications. In the labs and activities, you will incorporate the use of Wireshark®, the Windows utilities command netstat, and Packet Tracer to investigate these two protocols.

Chapter 5 – Chapter 5 introduces the OSI Network layer. You will examine concepts of addressing and routing and learn about path determination, data packets, and the IP protocol. By the end of this chapter, you will configure hosts to access the local network and explore routing tables.

Chapter 6 - In Chapter 6, you will focus on network addressing in detail and learn how to use the address mask, or prefix length, to determine the number of subnetworks and hosts in a network. You will also be introduced to ICMP (Internet Control Message Protocol) tools, such as ping and trace.

Chapter 7 – Chapter 7 discusses the services provided by Data Link layer. An emphasis is placed on the encapsulation processes that occur as data travels across the LAN and the WAN.

Chapter 8 – Chapter 8 introduces the Physical layer. You will discover how data sends signals and is encoded for travel across the network. You'll learn about bandwidth and also about the types of media and their associated connectors.

Chapter 9 – In Chapter 9, you will examine the technologies and operation of Ethernet. You will use Wireshark®, Packet Tracer activities, and lab exercises to explore Ethernet.

Chapter 10 – Chapter 10 focuses on designing and cabling a network. You will apply the knowledge and skills developed in the previous chapters to determine the appropriate cables to use, how to connect devices, and develop an addressing and testing scheme.

Chapter 11 –In Chapter 11, you will connect and configure a small network using basic Cisco IOS commands for routers and switches. Upon completion of this final chapter, you will be prepared you to go on to either CCNA Exploration Routing or CCNA Exploration Switching courses.

Wireshark®, is a registered trademark of Gerald Combs

Application Layer Functionality and Protocols

Chapter Introduction

Most of us experience the Internet through the World Wide Web, e-mail services, and file-sharing programs. These applications, and many others, provide the human interface to the underlying network, enabling us to send and receive information with relative ease. Typically the applications that we use are intuitive, meaning we can access and use them without knowing how they work. However, for network professionals, it is important to know how an application is able to format, transmit and interpret messages that are sent and received across the network.

Visualizing the mechanisms that enable communication across the network is made easier if we use the layered framework of the *Open System Interconnection (OSI)* model. In this chapter, we will focus on the role of one layer, the Application layer and its components: applications, services, and protocols. We will explore how these three elements make the robust communication across the information network possible.

In this chapter, you will learn to:

- Describe how the functions of the three upper OSI model layers provide network services to end user applications.

- Describe how the TCP/IP Application Layer protocols provide the services specified by the upper layers of the OSI model.

- Define how people use the Application Layer to communicate across the information network.

- Describe the function of well-known TCP/IP applications, such as the World Wide Web and email, and their related services (HTTP, DNS, SMB, DHCP, SMTP/POP, and Telnet).

- Describe file-sharing processes that use peer-to-peer applications and the Gnutella protocol.

- Explain how protocols ensure services running on one kind of device can send to and receive data from many different network devices.

- Use network analysis tools to examine and explain how common user applications work.

3.1 Applications - The Interface Between the Networks

3.1.1 OSI and TCP/IP Model

The Open Systems Interconnection reference model is a layered, abstract representation created as a guideline for network protocol design. The OSI model divides the networking process into seven logical layers, each of which has unique functionality and to which are assigned specific services and protocols.

In this model, information is passed from one layer to the next, starting at the Application layer on the transmitting host, proceeding down the hierarchy to the Physical layer, then passing over the communications channel to the destination host, where the information proceeds back up the hierarchy, ending at the Application layer. The figure depicts the steps in this process.

The Application layer, Layer seven, is the top layer of both the OSI and TCP/IP models. It is the layer that provides the interface between the applications we use to communicate and the underlying network over which our messages are transmitted. Application layer protocols are used to exchange data between programs running on the source and destination hosts. There are many Application layer protocols and new protocols are always being developed.

Although the TCP/IP protocol suite was developed prior to the definition of the OSI model, the functionality of the TCP/IP Application layer protocols fit roughly into the framework of the top three layers of the OSI model: Application, Presentation and Session layers.

Most TCP/IP Application layer protocols were developed before the emergence of personal computers, graphical user interfaces and multimedia objects. As a result, these protocols implement very little of the functionality that is specified in the OSI model Presentation and Session layers.

The Presentation Layer

The Presentation layer has three primary functions:

- Coding and conversion of Application layer data to ensure that data from the *source device* can be interpreted by the appropriate application on the destination device.

- Compression of the data in a manner that can be decompressed by the destination device.

- Encryption of the data for transmission and the decryption of data upon receipt by the destination.

Presentation layer implementations are not typically associated with a particular protocol stack. The standards for video and graphics are examples. Some well-known standards for video include QuickTime and Motion Picture Experts Group (MPEG). QuickTime is an Apple Computer specification for video and audio, and MPEG is a standard for video compression and coding.

Among the well-known graphic image formats are Graphics Interchange Format (GIF), Joint Photographic Experts Group (JPEG), and Tagged Image File Format (TIFF). GIF and JPEG are compression and coding standards for graphic images, and TIFF is a standard coding format for graphic images.

The Session Layer

As the name of the Session layer implies, functions at this layer create and maintain dialogs between source and destination applications. The Session layer handles the exchange of information to initiate dialogs, keep them active, and to restart sessions that are disrupted or idle for a long period of time.

Most applications, like web browsers or e-mail clients, incorporate functionality of the OSI layers 5, 6 and 7.

The most widely-known TCP/IP Application layer protocols are those that provide for the exchange of user information. These protocols specify the format and control information necessary for many of the common Internet communication functions. Among these TCP/IP protocols are:

- Domain Name Service Protocol (DNS) is used to resolve Internet names to IP addresses.

- Hypertext Transfer Protocol (HTTP) is used to transfer files that make up the Web pages of the World Wide Web.

- *Simple Mail Transfer Protocol (SMTP)* is used for the transfer of mail messages and attachments.

- Telnet, a terminal emulation protocol, is used to provide remote access to servers and networking devices.

- File Transfer Protocol (FTP) is used for interactive file transfer between systems.

The protocols in the TCP/IP suite are generally defined by Requests for Comments (RFCs). The Internet Engineering Task Force maintains the RFCs as the standards for the TCP/IP suite.

3.1.2 Application Layer Software

The functions associated with the Application layer protocols enable our human network to interface with the underlying data network. When we open a web browser or an instant message window, an application is started, and the program is put into the device's memory where it is executed. Each executing program loaded on a device is referred to as a process.

Within the Application layer, there are two forms of software programs or processes that provide access to the network: applications and services.

Network-Aware Applications

Applications are the software programs used by people to communicate over the network. Some end-user applications are network-aware, meaning that they implement the Application layer protocols and are able to communicate directly with the lower layers of the protocol stack. E-mail clients and web browsers are examples of these types of applications.

Application layer Services

Other programs may need the assistance of Application layer services to use network resources, like file transfer or network print spooling. Though transparent to the user, these services are the programs that interface with the network and prepare the data for transfer. Different types of data - whether it is text, graphics, or video - require different network services to ensure that it is properly prepared for processing by the functions occurring at the lower layers of OSI model.

Each application or network service uses protocols which define the standards and data formats to be used. Without protocols, the data network would not have a common way to format and direct data. In order to understand the function of various network services, it is necessary to become familiar with the underlying protocols that govern their operation.

Roll over the buttons in the figure to view examples.

3.1.3 User Applications, Services, and Application Layer Protocols

As mentioned previously, the Application layer uses protocols that are implemented within applications and services. While applications provide people with a way to create messages and Application layer services establish an interface to the network, protocols provide the rules and formats that govern how data is treated. All three components may be used by a single executable program and may even use the same name. For example, when discussing "Telnet" we could be referring to the application, the service, or the protocol.

In the OSI model, applications that interact directly with people are considered to be at the top of the stack, as are the people themselves. Like all layers within the OSI model, the Application layer relies on the functions of the lower layers in order to complete the communication process. Within

the Application layer, protocols specify what messages are exchanged between the source and destination hosts, the *syntax* of the control commands, the type and format of the data being transmitted, and the appropriate methods for error notification and recovery.

Play the animation to see the interaction between applications, services, and protocols.

3.1.4 Application Layer Protocol Functions

Application layer protocols are used by both the source and destination devices during a communication *session*. In order for the communications to be successful, the Application layer protocols implemented on the source and destination host must match.

Protocols establish consistent rules for exchanging data between applications and services loaded on the participating devices. Protocols specify how data inside the messages is structured and the types of messages that are sent between source and destination. These messages can be requests for services, acknowledgments, data messages, status messages, or error messages. Protocols also define message dialogues, ensuring that a message being sent is met by the expected response and the correct services are invoked when data transfer occurs.

Many different types of applications communicate across data networks. Therefore, Application layer services must implement multiple protocols to provide the desired range of communication experiences. Each protocol has a specific purpose and contains the characteristics required to meet that purpose. The right protocol details in each layer must be followed so that the functions at one layer interface properly with the services in the lower layer.

Applications and services may also use multiple protocols in the course of a single conversation. One protocol may specify how to establish the network connection and another describe the process for the data transfer when the message is passed to the next lower layer.

3.2 Making Provisions for Applications and Services

3.2.1 The Client-Server Model

When people attempt to access information on their device, whether it is a PC, laptop, PDA, cell phone, or some other device connected to a network, the data may not be physically stored on their device. If that is the case, a request to access that information must be made to the device where the data resides.

The Client/Server model

In the *client/server model*, the device requesting the information is called a client and the device responding to the request is called a server. Client and server processes are considered to be in the Application layer. The client begins the exchange by requesting data from the server, which responds by sending one or more streams of data to the client. Application layer protocols describe the format of the requests and responses between clients and servers. In addition to the actual data transfer, this exchange may also require control information, such as user authentication and the identification of a data file to be transferred.

One example of a client/server network is a corporate environment where employees use a company e-mail server to send, receive and store e-mail. The e-mail client on an employee computer issues a request to the e-mail server for any unread mail. The server responds by sending the requested e-mail to the client.

Although data is typically described as flowing from the server to the client, some data always flows from the client to the server. Data flow may be equal in both directions, or may even be greater in the direction going from the client to the server. For example, a client may transfer a file to the server for storage purposes. **Data transfer from a client to a server is referred to as an** *upload* and data from a server to a client as a download.

Roll over the tabs in the figure to view file transfer.

3.2.2 Servers

In a general networking context, any device that responds to requests from client applications is functioning as a server. A server is usually a computer that contains information to be shared with many client systems. For example, web pages, documents, databases, pictures, video, and audio files can all be stored on a server and delivered to requesting clients. In other cases, such as a network printer, the print server delivers the client print requests to the specified printer.

Different types of server applications may have different requirements for client access. Some servers may require authentication of user account information to verify if the user has permission to access the requested data or to use a particular operation. Such servers rely on a central list of user accounts and the authorizations, or permissions, (both for data access and operations) granted to each user. When using an FTP client, for example, if you request to upload data to the FTP server, you may have permission to write to your individual folder but not to read other files on the site.

In a client/server network, the server runs a service, or process, sometimes called a server *daemon*. Like most services, daemons typically run in the background and are not under an end user's direct control. Daemons are described as "listening" for a request from a client, because they are programmed to respond whenever the server receives a request for the service provided by the daemon. When a daemon "hears" a request from a client, it exchanges appropriate messages with the client, as required by its protocol, and proceeds to send the requested data to the client in the proper format.

3.2.3 Application Layer Services and Protocols

A single application may employ many different supporting Application layer services; thus what appears to the user as one request for a web page may, in fact, amount to dozens of individual requests. And for each request, multiple processes may be executed. For example, a client may require several individual processes to formulate just one request to a server.

Additionally, servers typically have multiple clients requesting information at the same time. For example, a **Telnet** server may have many clients requesting connections to it. These individual client requests must be handled simultaneously and separately for the network to succeed. The Application layer processes and services rely on support from lower layer functions to successfully manage the multiple conversations.

In this activity, you will study a simple example of client-server interaction, which can serve as a model for more complex interactions later in the course.

3.2.4 Peer-to-Peer Networking and Applications (P2P)

The Peer-to-Peer Model

In addition to the client/server model for networking, there is also a peer-to-peer model. Peer-to-peer networking involves two distinct forms: peer-to-peer network design and peer-to-peer applications (P2P). Both forms have similar features but in practice work very differently.

Peer-to-Peer Networks

In a peer-to-peer network, two or more computers are connected via a network and can share re-sources (such as printers and files) without having a dedicated server. Every connected end device (known as a peer) can function as either a server or a client. One computer might assume the role of server for one transaction while simultaneously serving as a client for another. The roles of client and server are set on a per request basis.

A simple home network with two connected computers sharing a printer is an example of a peer-to-peer network. Each person can set his or her computer to share files, enable networked games, or share an Internet connection. Another example of peer-to-peer network functionality is two computers connected to a large network that use software applications to share resources between one another through the network.

Unlike the client/server model, which uses dedicated servers, peer-to-peer networks decentralize the resources on a network. Instead of locating information to be shared on dedicated servers, in-formation can be located anywhere on any connected device. Most of the current operating sys-tems support file and print sharing without requiring additional server software. Because peer-to-peer networks usually do not use centralized user accounts, permissions, or monitors, it is difficult to enforce security and access policies in networks containing more than just a few com-puters. User accounts and access rights must be set individually on each peer device.

Peer-to-Peer Applications

A peer-to-peer application (P2P), unlike a peer-to-peer network, allows a device to act as both a client and a server within the same communication. In this model, every client is a server and every server a client. Both can initiate a communication and are considered equal in the communi-cation process. However, peer-to-peer applications require that each end device provide a user in-terface and run a background service. When you launch a specific peer-to-peer application it invokes the required user interface and background services. After that the devices can communi-cate directly.

Some P2P applications use a hybrid system where resource sharing is decentralized but the in-dexes that point to resource locations are stored in a centralized directory. In a hybrid system, each peer accesses an index server to get the location of a resource stored on another peer. The index server can also help connect two peers, but once connected, the communication takes place be-tween the two peers without additional communication to the index server.

Peer-to-peer applications can be used on peer-to-peer networks, client/server networks, and across the Internet.

3.3 Application Layer Protocols and Services Examples

3.3.1 DNS Services and Protocol

Now that we have a better understanding of how applications provide an interface for the user and provide access to the network, we will take a look at some specific commonly used protocols.

As we will see later in this course, the Transport layer uses an addressing *scheme* called a port number. Port numbers identify applications and Application layer services that are the source and destination of data. Server programs generally use predefined port numbers that are commonly known by clients. As we examine the different TCP/IP Application layer protocols and services,

we will be referring to the TCP and UDP port numbers normally associated with these services. Some of these services are:

- *Domain Name System (DNS)* - TCP/UDP Port 53
- Hypertext Transfer Protocol (HTTP) - TCP Port 80
- Simple Mail Transfer Protocol (SMTP) - TCP Port 25
- *Post Office Protocol (POP)* - TCP Port 110
- Telnet - TCP Port 23
- Dynamic Host Configuration Protocol - UDP Ports 67 and 68
- File Transfer Protocol (FTP) - TCP Ports 20 and 21

DNS

In data networks, devices are labeled with numeric IP addresses, so that they can participate in sending and receiving messages over the network. However, most people have a hard time remembering this numeric address. Hence, domain names were created to convert the numeric address into a simple, recognizable name.

On the Internet these domain names, such as www.cisco.com, are much easier for people to remember than 198.133.219.25, which is the actual numeric address for this server. Also, if Cisco decides to change the numeric address, it is transparent to the user, since the domain name will remain www.cisco.com. The new address will simply be linked to the existing domain name and connectivity is maintained. When networks were small, it was a simple task to maintain the mapping between domain names and the addresses they represented. However, as networks began to grow and the number of devices increased, this manual system became unworkable.

The Domain Name System (DNS) was created for domain name to address resolution for these networks. DNS uses a distributed set of servers to resolve the names associated with these numbered addresses.

The **DNS protocol** defines an automated service that matches resource names with the required numeric *network address*. It includes the format for queries, responses, and data formats. DNS protocol communications use a single format called a message. This message format is used for all types of client queries and server responses, error messages, and the transfer of *resource record* information between servers.

DNS is a client/server service; however, it differs from the other client/server services that we are examining. While other services use a client that is an application (such as web browser, e-mail client), the DNS client runs as a service itself. The DNS client, sometimes called the *DNS resolver*, supports name resolution for our other network applications and other services that need it.

When configuring a network device, we generally provide one or more **DNS Server** addresses that the DNS client can use for name resolution. Usually the Internet service provider provides the addresses to use for the DNS servers. When a user's application requests to connect to a remote device by name, the requesting DNS client queries one of these name servers to resolve the name to a numeric address.

Computer operating systems also have a utility called `nslookup` **that allows the user to manually *query*** the name servers to resolve a given host name. This utility can also be used to troubleshoot name resolution issues and to verify the current status of the name servers.

In the figure, when the `nslookup` is issued, the default DNS server configured for your host is displayed. In this example, the DNS server is dns-sjk.cisco.com which has an address of 171.68.226.120.

We then can type the name of a host or domain for which we wish to get the address. In the first query in the figure, a query is made for www.cisco.com. The responding name server provides the address of 198.133.219.25.

The queries shown in the figure are only simple tests. The `nslookup` has many options available for extensive testing and verification of the DNS process.

A DNS server provides the name resolution using the name daemon, which is often called named, (pronounced name-dee).

The DNS server stores different types of resource records used to resolve names. These records contain the name, address, and type of record.

Some of these record types are:

- A - an end device address

- NS - an *authoritative* name server

- CNAME - the canonical name (or Fully Qualified Domain Name) for an alias; used when multiple services have the single network address but each service has its own entry in DNS

- MX - mail exchange record; maps a *domain name* to a list of mail exchange servers for that domain

When a client makes a query, the server's "named" process first looks at its own records to see if it can resolve the name. If it is unable to resolve the name using its stored records, it contacts other servers in order to resolve the name.

The request may be passed along to a number of servers, which can take extra time and consume bandwidth. Once a match is found and returned to the original requesting server, the server temporarily stores the numbered address that matches the name in *cache*.

If that same name is requested again, the first server can return the address by using the value stored in its name cache. Caching reduces both the DNS query data network traffic and the workloads of servers higher up the hierarchy. The DNS Client service on Windows PCs optimizes the performance of DNS name resolution by storing previously resolved names in memory, as well. The `ipconfig /displaydns` command displays all of the cached DNS entries on a Windows XP or 2000 computer system.

The Domain Name System uses a hierarchical system to create a name database to provide name resolution. The hierarchy looks like an inverted tree with the root at the top and branches below.

At the top of the hierarchy, the root servers maintain records about how to reach the top-level domain servers, which in turn have records that point to the secondary level domain servers and so on.

The different top-level domains represent either the type of organization or the country of origin. Examples of top-level domains are:

- **.au - Australia**

- **.co - Colombia**

- **.com - a business or industry**

- **.jp - Japan**

- **.org - a non-profit organization**

After top-level domains are second-level domain names, and below them are other lower level domains.

Each domain name is a path down this inverted tree starting from the root.

For example, as shown in the figure, the root DNS server may not know exactly where the e-mail server mail.cisco.com is located, but it maintains a record for the "com" domain within the top-level domain. Likewise, the servers within the "com" domain may not have a record for mail.cisco.com, but they do have a record for the "cisco.com" domain. The servers within the cisco.com domain have a record (a MX record to be precise) for mail.cisco.com.

The Domain Name System relies on this hierarchy of decentralized servers to store and maintain these resource records. The resource records list domain names that the server can resolve and alternative servers that can also process requests. If a given server has resource records that correspond to its level in the domain hierarchy, it is said to be **authoritative** for those records.

For example, a name server in the cisco.netacad.net domain would not be authoritative for the mail.cisco.com record because that record is held at a higher domain level server, specifically the name server in the cisco.com domain.

Links

> http://www.ietf.org//rfc/rfc1034.txt

> http://www.ietf.org/rfc/rfc1035.txt

3.3.2 WWW Service and HTTP

When a web address (or URL) is typed into a web browser, the web browser establishes a connection to the web service running on the server using the HTTP protocol. URLs (or Uniform Resource Locator) and URIs (Uniform Resource Identifier) are the names most people associate with web addresses.

The URL http://www.cisco.com/index.html is an example of a URL that refers to a specific resource - a web page named **index.html** on a server identified as **cisco.com** (click the tabs in the figure to see the steps used by HTTP).

Web browsers are the client applications our computers use to connect to the World Wide Web and access resources stored on a web server. As with most server processes, the web server runs as a background service and makes different types of files available.

In order to access the content, web clients make connections to the server and request the desired resources. The server replies with the resources and, upon receipt, the browser interprets the data and presents it to the user.

Browsers can interpret and present many data types, such as plain text or Hypertext Markup Language (HTML, the language in which web pages are constructed). Other types of data, however, may require another service or program, typically referred to as plug-ins or add-ons. To help the browser determine what type of file it is receiving, the server specifies what kind of data the file contains.

To better understand how the web browser and web client interact, we can examine how a web page is opened in a browser. For this example, we will use the URL: http://www.cisco.com/web-server.htm.

First, the browser interprets the three parts of the URL:

Step 1. **http** (the protocol or scheme)

Step 2. www.cisco.com (the server name)

Step 3. **web-server.htm** (the specific file name requested).

The browser then checks with a name server to convert www.cisco.com into a numeric address, which it uses to connect to the server. Using the HTTP protocol requirements, the browser sends a GET request to the server and asks for the file **web-server.htm**. The server in turn sends the HTML *code* for this web page to the browser. Finally, the browser deciphers the HTML code and formats the page for the browser window.

The Hypertext Transfer Protocol (HTTP), one of the protocols in the TCP/IP suite, was originally developed to publish and retrieve HTML pages and is now used for distributed, collaborative information systems. HTTP is used across the World Wide Web for data transfer and is one of the most used application protocols.

HTTP specifies a request/response protocol. When a client, typically a web browser, sends a request message to a server, the HTTP protocol defines the message types the client uses to request the web page and also the message types the server uses to respond. The three common message types are GET, POST, and PUT.

GET is a client request for data. A web browser sends the **GET** message to request pages from a web server. As shown in the figure, once the server receives the **GET** request, it responds with a status line, such as HTTP/1.1 200 OK, and a message of its own, the body of which may be the requested file, an error message, or some other information.

POST and **PUT** are used to send messages that upload data to the web server. For example, when the user enters data into a form embedded in a web page, **POST** includes the data in the message sent to the server.

PUT uploads resources or content to the web server.

Although it is remarkably flexible, HTTP is not a secure protocol. The **POST** messages upload information to the server in plain text that can be intercepted and read. Similarly, the server responses, typically HTML pages, are also unencrypted.

For secure communication across the Internet, the HTTP Secure (HTTPS) protocol is used for accessing or posting web server information. HTTPS can use authentication and *encryption* to secure data as it travels between the client and server. HTTPS specifies additional rules for passing data between the Application layer and the Transport Layer.

In this activity, you will configure DNS and HTTP services, and then study the packets that result when a web page is requested by typing a URL.

3.3.3 E-mail Services and SMTP/POP Protocols

E-mail, the most popular network service, has revolutionized how people communicate through its simplicity and speed. Yet to run on a computer or other end device, e-mail requires several applications and services. Two example Application layer protocols are Post Office Protocol (POP) and **Simple Mail Transfer Protocol** (SMTP), shown in the figure. As with HTTP, these protocols define client/server processes.

When people compose e-mail messages, they typically use an application called a *Mail User Agent (MUA)*, or e-mail client. The MUA allows messages to be sent and places received messages into the client's mailbox, both of which are distinct processes.

In order to receive e-mail messages from an e-mail server, the e-mail client can use POP. Sending e-mail from either a client or a server uses message formats and command strings defined by the SMTP protocol. Usually an e-mail client provides the functionality of both protocols within one application.

E-mail Server Processes - MTA and MDA

The e-mail server operates two separate processes:

- Mail Transfer Agent (MTA)

- Mail Delivery Agent (MDA)

The Mail Transfer Agent (MTA) process is used to forward e-mail. As shown in the figure, the MTA receives messages from the MUA or from another MTA on another e-mail server. Based on the message header, it determines how a message has to be forwarded to reach its destination. If the mail is addressed to a user whose mailbox is on the local server, the mail is passed to the MDA. If the mail is for a user not on the local server, the MTA routes the e-mail to the MTA on the appropriate server.

In the figure, we see that the Mail Delivery Agent (MDA) accepts a piece of e-mail from a Mail Transfer Agent (MTA) and performs the actual delivery. The MDA receives all the inbound mail from the MTA and places it into the appropriate users' mailboxes. The MDA can also resolve final delivery issues, such as virus scanning, *spam filtering*, and return-receipt handling. Most e-mail communications use the MUA, MTA, and MDA applications. However, there are other alternatives for e-mail delivery.

A client may be connected to a corporate e-mail system, such as IBM's Lotus Notes, Novell's Groupwise, or Microsoft's Exchange. These systems often have their own internal e-mail format, and their clients typically communicate with the e-mail server using a proprietary protocol. The server sends or receives e-mail via the Internet through the product's Internet mail *gateway*, which performs any necessary reformatting.

As another alternative, computers that do not have an MUA can still connect to a mail service on a web browser in order to retrieve and send messages in this manner. Some computers may run their own MTA and manage inter-domain e-mail themselves. If, for example, two people who work for the same company exchange e-mail with each other using a proprietary protocol, their messages may stay completely within the company's corporate e-mail system.

As mentioned earlier, e-mail can use the protocols, POP and SMTP (see the figure for an explanation of how they each work). POP and POP3 (Post Office Protocol, version 3) are inbound mail **delivery** protocols and are typical client/server protocols. They deliver e-mail from the e-mail server to the client (MUA). The MDA listens for when a client connects to a server. Once a connection is established, the server can deliver the e-mail to the client.

The Simple Mail Transfer Protocol (SMTP), on the other hand, governs the transfer of **outbound** e-mail from the sending client to the e-mail server (MDA), as well as the transport of e-mail between e-mail servers (MTA). SMTP enables e-mail to be transported across data networks between different types of server and client software and makes e-mail exchange over the Internet possible.

The SMTP protocol message format uses a rigid set of commands and replies. These commands support the procedures used in SMTP, such as session initiation, mail transaction, forwarding mail, verifying mailbox names, expanding mailing lists, and the opening and closing exchanges.

Some of the commands specified in the SMTP protocol are:

- HELO - identifies the SMTP client process to the SMTP server process

- EHLO - Is a newer version of HELO, which includes services extensions

- MAIL FROM - Identifies the sender

- RCPT TO - Identifies the recipient
- DATA - Identifies the body of the message

3.3.4 FTP

The File Transfer Protocol (FTP) is another commonly used Application layer protocol. FTP was developed to allow for file transfers between a client and a server. An FTP client is an application that runs on a computer that is used to push and pull files from a server running the FTP daemon (FTPd).

To successfully transfer files, FTP requires two connections between the client and the server: one for commands and replies, the other for the actual file transfer.

The client establishes the first connection to the server on TCP port 21. This connection is used for control traffic, consisting of client commands and server replies.

The client establishes the second connection to the server over TCP port 20. This connection is for the actual file transfer and is created every time there is a file transferred.

The file transfer can happen in either direction. The client can download (pull) a file from the server or, the client can upload (push) a file to the server.

3.3.5 DHCP

The *Dynamic Host Configuration Protocol (DHCP)* service enables devices on a network to obtain IP addresses and other information from a DHCP server. This service automates the assignment of IP addresses, subnet masks, gateway and other IP networking parameters.

DHCP allows a host to obtain an IP address dynamically when it connects to the network. The DHCP server is contacted and an address requested. The DHCP server chooses an address from a configured range of addresses called a pool and assigns ("leases") it to the host for a set period.

On larger local networks, or where the user population changes frequently, DHCP is preferred. New users may arrive with laptops and need a connection. Others have new workstations that need to be connected. Rather than have the network administrator assign IP addresses for each workstation, it is more efficient to have IP addresses assigned automatically using DHCP.

DHCP distributed addresses are not permanently assigned to hosts but are only leased for a period of time. If the host is powered down or taken off the network, the address is returned to the pool for reuse. This is especially helpful with mobile users that come and go on a network. Users can freely move from location to location and re-establish network connections. The host can obtain an IP address once the hardware connection is made, either via a wired or wireless LAN.

DHCP makes it possible for you to access the Internet using wireless hotspots at airports or coffee shops. As you enter the area, your laptop DHCP client contacts the local DHCP server via a wireless connection. The DHCP server assigns an IP address to your laptop.

As the figure shows, various types of devices can be DHCP servers when running DHCP service software. The DHCP server in most medium to large networks is usually a local dedicated PC-based server.

With home networks the DHCP server is usually located at the ISP and a host on the home network receives its IP configuration directly from the ISP.

DHCP can pose a security risk because any device connected to the network can receive an address. This risk makes physical security an important factor when determining whether to use dynamic or manual addressing.

Dynamic and static addressing both have their places in network designs. Many networks use both DHCP and static addressing. DHCP is used for general purpose hosts such as end user devices, and fixed addresses are used for network devices such as gateways, switches, servers and printers.

Without DHCP, users have to manually input the IP address, *subnet mask* and other network settings in order to join the network. The DHCP server maintains a pool of IP addresses and leases an address to any DHCP-enabled client when the client is powered on. Because the IP addresses are dynamic (leased) rather than static (permanently assigned), addresses no longer in use are automatically returned to the pool for reallocation. When a DHCP-configured device boots up or connects to the network, the client broadcasts a DHCP DISCOVER packet to identify any available DHCP servers on the network. A DHCP server replies with a DHCP OFFER, which is a lease offer message with an assigned IP address, subnet mask, DNS server, and *default gateway* information as well as the duration of the lease.

The client may receive multiple DHCP OFFER packets if there is more than one DHCP server on the local network, so it must choose between them, and broadcast a DHCP REQUEST packet that identifies the explicit server and lease offer that the client is accepting. A client may choose to request an address that it had previously been allocated by the server.

Assuming that the IP address requested by the client, or offered by the server, is still valid, the server would return a DHCP *ACK* message that acknowledges to the client the lease is finalized. If the offer is no longer valid - perhaps due to a time-out or another client allocating the lease - then the selected server will respond with a DHCP NAK message (Negative Acknowledgement). If a DHCP NAK message is returned, then the selection process must begin again with a new DHCP DISCOVER message being transmitted.

Once the client has the lease, it must be renewed prior to the lease expiration through another DHCP REQUEST message.

The DHCP server ensures that all IP addresses are unique (an IP address cannot be assigned to two different network devices simultaneously). Using DHCP enables network administrators to easily reconfigure client IP addresses without having to manually make changes to the clients. Most Internet providers use DHCP to allocate addresses to their customers who do not require a static address.

The fourth *CCNA* Exploration course will cover the operation of DHCP in greater detail.

3.3.6 File Sharing Services and SMB Protocol

The *Server Message Block (SMB)* is a client/server file sharing protocol. IBM developed Server Message Block (SMB) in the late 1980s to describe the structure of shared network resources, such as directories, files, printers, and serial ports. It is a request-response protocol. Unlike the file sharing supported by FTP, clients establish a long term connection to servers. Once the connection is established, the user of the client can access the resources on the server as if the resource is local to the client host.

SMB file-sharing and print services have become the mainstay of Microsoft networking. With the introduction of the Windows 2000 series of software, Microsoft changed the underlying structure for using SMB. In previous versions of Microsoft products, the SMB services used a non-TCP/IP protocol to implement name resolution. Beginning with Windows 2000, all subsequent Microsoft products use DNS naming. This allows TCP/IP protocols to directly support SMB resource sharing, as shown in the figure.

The LINUX and *UNIX* operating systems also provide a method of sharing resources with Microsoft networks using a version of SMB called SAMBA. The Apple Macintosh operating systems also support resource sharing using the SMB protocol.

The SMB protocol describes file system access and how clients can make requests for files. It also describes the SMB protocol inter-process communication. All SMB messages share a common format. This format uses a fixed-sized header followed by a variable-sized parameter and data component.

SMB messages can:

- Start, authenticate, and terminate sessions
- Control file and printer access
- Allow an application to send or receive messages to or from another device

The SMB file exchange process is shown in the figure.

3.3.7 P2P Services and Gnutella Protocol

You learned about FTP and SMB as ways of obtaining files, here is another Application protocol. Sharing files over the Internet has become extremely popular. With **P2P applications** based on the Gnutella protocol, people can make files on their hard disks available to others for downloading. Gnutella-compatible client software allows users to connect to Gnutella services over the Internet and to locate and access resources shared by other Gnutella peers.

Many client applications are available for accessing the Gnutella network, including: BearShare, Gnucleus, LimeWire, Morpheus, WinMX and XoloX (see a screen capture of LimeWire in the figure). While the Gnutella Developer Forum maintains the basic protocol, application vendors often develop extensions to make the protocol work better on their applications.

Many P2P applications do not use a central database to record all the files available on the peers. Instead, the devices on the network each tell the other what files are available when queried and use the Gnutella protocol and services to support locating resources. See the figure.

When a user is connected to a Gnutella service, the client applications will search for other Gnutella nodes to connect to. These nodes handle queries for resource locations and replies to those requests. They also govern control messages, which help the service discover other nodes. The actual file transfers usually rely on HTTP services.

The Gnutella protocol defines five different packet types:

- ping - for device discovery
- pong - as a reply to a ping
- query - for file location
- query hit - as a reply to a query
- push - as a download request

3.3.8 Telnet Services and Protocol

Long before desktop computers with sophisticated graphical interfaces existed, people used text-based systems which were often just display terminals physically attached to a central computer. Once networks were available, people needed a way to remotely access the computer systems in the same manner that they did with the directly attached terminals.

Telnet was developed to meet that need. Telnet dates back to the early 1970s and is among the oldest of the Application layer protocols and services in the TCP/IP suite. Telnet provides a standard

method of emulating text-based terminal devices over the data network. Both the protocol itself and the client software that implements the protocol are commonly referred to as Telnet.

Appropriately enough, a connection using Telnet is called a Virtual Terminal (VTY) session, or connection. Rather than using a physical device to connect to the server, Telnet uses software to create a virtual device that provides the same features of a terminal session with access to the server command line interface (CLI).

To support Telnet client connections, the server runs a service called the Telnet daemon. A virtual terminal connection is established from an end device using a Telnet client application. Most operating systems include an Application layer Telnet client. On a Microsoft Windows PC, Telnet can be run from the command prompt. Other common terminal applications that run as Telnet clients are HyperTerminal, Minicom, and TeraTerm.

Once a Telnet connection is established, users can perform any authorized function on the server, just as if they were using a command line session on the server itself. If authorized, they can start and stop processes, configure the device, and even shut down the system.

Click the tabs in the figure to view the Telnet example.

Refer to
Figure
in online course

Telnet is a client/server protocol and it specifies how a VTY session is established and terminated. It also provides the syntax and order of the commands used to initiate the Telnet session, as well as control commands that can be issued during a session. Each Telnet command consists of at least two bytes. The first byte is a special character called the **Interpret as Command** (IAC) character. As its name implies, the IAC defines the next byte as a command rather than text.

Some sample Telnet protocol commands include:

Are You There (AYT) - Lets the user request that something appear on the terminal screen to indicate that the VTY session is active.

Erase Line (EL) - Deletes all text from the current line.

Interrupt Process (IP) - Suspends, interrupts, aborts, or terminates the process to which the Virtual Terminal is connected. For example, if a user started a program on the Telnet server via the VTY, he or she could send an IP command to stop the program.

While the Telnet protocol supports user authentication, it does not support the transport of *encrypted* data. All data exchanged during a Telnet sessions is transported as plain text across the network. This means that the data can be intercepted and easily understood.

If security is a concern, the Secure Shell (SSH) protocol offers an alternate and secure method for server access. SSH provides the structure for secure remote login and other secure network services. It also provides stronger authentication than Telnet and supports the transport of session data using encryption. **As a best practice, network professionals should always use SSH in place of Telnet, whenever possible.**

Later in this course, we will use Telnet and SSH to access and configure network devices over the lab network.

3.4 Chapter Labs and Activities

3.4.1 Data Stream Capture

Refer to
Lab Activity
for this chapter

In this activity, you will use a computer that has a microphone and Microsoft Sound Recorder or Internet access so that an audio file can be downloaded.

3.4.2 Lab - Managing a Web Server

In this lab you will download, install, and configure the popular Apache web server. A web browser will be used to connect to the server, and Wireshark will be used to capture the communication. Analysis of the capture will aid students in understanding how the HTTP protocol operates.

3.4.3 Lab - E-mail Services and Protocols

In this lab, you will configure and use an e-mail client application to connect to eagle-server network services. You will then monitor the communication with Wireshark and analyze the captured packets.

Summary and Review

The **Application layer** is responsible for directly accessing the underlying processes that manage and deliver communication to the human network. This layer serves as the source and destination of communications across data networks.

The Application layer applications, protocols, and services enable users to interact with the data network in a way that is meaningful and effective.

Applications are computer programs with which the user interacts and which initiate the data transfer process at the user's request.

Services are background programs that provide the connection between the Application layer and the lower layers of the networking model.

Protocols provide a structure of agreed-upon rules and processes that ensure services running on one particular device can send and receive data from a range of different network devices.

Delivery of data over the network can be requested from a **server** by a **client**, or between devices that operate in a **peer-to-peer** arrangement, where the client/server relationship is established according to which device is the source and destination at that time. **Messages** are exchanged between the Application layer services at each end device in accordance with the protocol specifications to establish and use these relationships.

Protocols like **HTTP**, for example, support the delivery of web pages to end devices. **SMTP/POP** protocols support sending and receiving e-mail. **SMB** enables users to share files. **DNS** resolves the human legible names used to refer to network resources into numeric addresses usable by the network.

In this activity, you will build and analyze more complex parts of the model of the Exploration lab network.

Packet Tracer Skills Integration Instructions (PDF)

To Learn More

Reflection Questions

Why is it important to distinguish between a particular Application layer application, the associated service, and the protocol? Discuss this in the context of network reference models.

What if it was possible to include all Application layer services with a single all-encompassing protocol? Discuss the advantages and disadvantages of having one such protocol.

How would you go about developing a new protocol for a new Application layer service? What would have to be included? Who would have to be involved in the process and how would the information be disseminated?

Links

```
http://www.ietf.org/
http://www.protocols.com/
```

Chapter Quiz

Take the chapter quiz to test your knowledge.

Your Chapter Notes