UNIT V – Application Layer

WWW and HTTP – FTP – Email –Telnet –SSH – DNS – SNMP.

World Wide Web (WWW)

- The Web is a repository of information in which the documents, called **web pages,** are distributed all over the world and related documents are linked together.
- The linking of web pages was achieved using a concept called **hypertext**.
- The term hypertext, has been changed to **hypermedia,** to show that a web page can be a text document, an image, an audio file, or a video file.

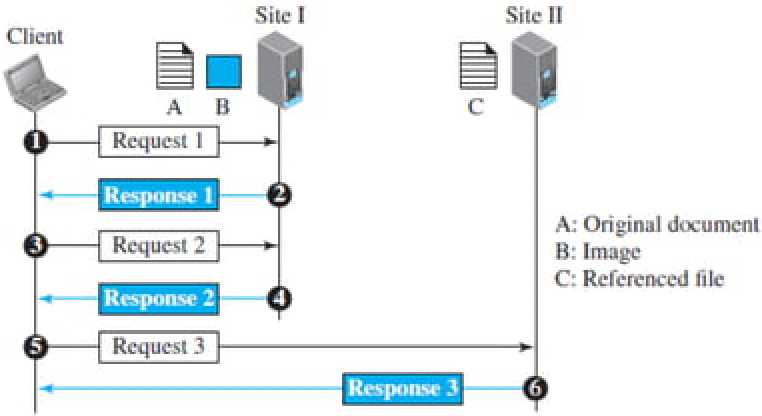
Architecture

- The WWW is a distributed client-server service, in which a client using a browser can access a service using a server.
- The service provided is distributed over many locations called **sites**.
- Each site holds one or more web pages.
- Each web page, can contain some links to other web pages in the same or other sites.

World Wide Web (WWW)

Architecture

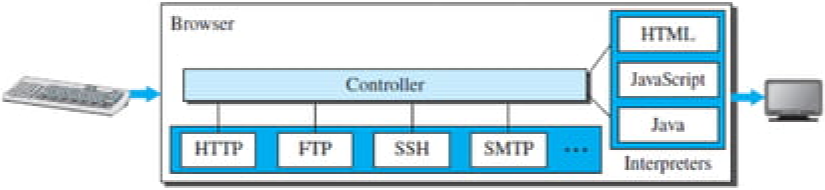
- In other words, **a web page** can be **simple or composite.**
- A **simple web page** has no links to other web pages; a **composite web page** has one or more links to other web pages.
- Each web page is a file with a name and address.



World Wide Web (WWW)

Web Client (Browser)

- **Browsers** interpret and display a web page.
- Each browser usually consists of three parts: **a controller, client protocols, and interpreters**.
- The **controller receives input** from the keyboard or the mouse and **uses the client programs to access the document**.
- After the document has been accessed, the **controller uses one of the interpreters to display the document** on the screen.
- The client protocol can be one of the protocols such as **HTTP or FTP**.
- The **interpreter** can be **HTML, Java, or JavaScript**, depending on the type of document.
- Some commercial browsers include Internet Explorer, Netscape Navigator, and Firefox.



World Wide Web (WWW)

Web Server

- The web page is stored at the server.
- Each time a request arrives, the corresponding document is sent to the client.
- To improve efficiency, servers normally store requested **files in a cache** in memory; memory is faster to access than a disk.
- A server can also become **more efficient** through **multithreading or multiprocessing**.
- In this case, a server can answer more than one request at a time.
- Some popular web servers include **Apache** and **Microsoft Internet Information Server**.

World Wide Web (WWW)

Uniform Resource Locator (URL)

- A web page, as a file, needs to have a unique identifier to distinguish it from other web pages.
- To define a web page, three identifiers are needed: **host, port, and path**.
- **Protocol:** The first identifier is the abbreviation for the client-server program to access the web page. Eg: HTTP, FTP.
- **Host:** It can be the IP address of the server or the unique name given to the server.
- **Port:** It is a 16-bit integer, is normally predefined for the client-server application.
- **Path:** It identifies the location and the name of the file in the underlying operating system. The format of this identifier normally depends on the operating system.
- **Example:**

– protocol://host/path	Used most of the time
– protocol://host:port/path	Used when port number is needed

World Wide Web (WWW)

Web Documents

- The documents in the WWW can be grouped into three broad categories: static, dynamic, and active.
- **Static Documents**
  - Fixed-content documents that are created and stored in a server.
  - The client can get a copy of the document only.
  - The contents in the server can be changed, but the user cannot change them.
  - Static documents are prepared using one of several languages: HyperText Markup Language (HTML), Extensible Markup Language (XML), Extensible Style Language (XSL), and Extensible Hypertext Markup Language (XHTML).

World Wide Web (WWW)

Web Documents

- **Dynamic Documents**
  - It is created by a web server whenever a browser requests the document.
  - When a request arrives, the web server runs an application program or a script that creates the dynamic document.
  - The server returns the result of the program or script as a response to the browser that requested the document.
  - A very simple example of a dynamic document is the retrieval of the time and date from a server.
  - Although the Common Gateway Interface (CGI) was used to retrieve a dynamic document.
  - The scripting languages such as Java Server Pages (JSP), which uses the Java language for scripting, or Active Server Pages (ASP), a Microsoft product that uses Visual Basic language for scripting, or ColdFusion, which embeds queries in a Structured Query Language (SQL) database in the HTML document.

World Wide Web (WWW)

Web Documents

- **Active Documents**
  - For many applications, a program or a script needs to be run at the client site.
  - These are called **active documents.**
  - For example, suppose we want to run a program that creates animated graphics on the screen or a program that interacts with the user.
  - The program definitely needs to be run at the client site where the animation or interaction takes place.
  - When a browser requests an active document, the server sends a copy of the document or a script.
  - The document is then run at the client (browser) site.
  - One way to create an active document is to use Java applets.

HyperText Transfer Protocol (HTTP)

- It is an application protocol that is used to retrieve Web pages from remote servers.
- All the web browsers use HTTP protocol to communicate with Web servers over the Internet.
- The main goal of the Web is to organize and retrieve information over the internet.
  - HyperText (Interlinked documents) is used for this purpose.
  - Hypertext is a document can link to another document
    - HTTP – Protocol used to retrieve hypertext
    - HTML – Document language or Markup language used to create hypertext
- To organize information into a system of linked documents or objects, we need to be able to retrieve one document to get started.

HyperText Transfer Protocol (HTTP)

- To organize information into a system of linked documents or objects, we need to be able to retrieve one document to get started.
- A hypertext document can be retrived by "opening a URL"
- **URL:** Location of a resource on the internet.
- Example: **http://www.cs.princeton.edu/index.html**
- Web browser would open a TCP connection to the Web server at a machine called **www.cs.princeton.edu** and immediately retrieve and display the file called **index.html**.
- Most files on the Web contain images and text and many have other objects such as audio and video clips, pieces of code, etc.
- They also frequently include URLs that point to other files that may be located on other machines, which is the core of the "hypertext" part of HTTP and HTML.

HyperText Transfer Protocol (HTTP)

- When a user select a page to view, browser (the client) fetches the page from the server using HTTP running over TCP.
- HTTP is a request/response protocol, where every message has the general form

START_LINE <CRLF>

MESSAGE_HEADER <CRLF>

<CRLF>

MESSAGE_BODY <CRLF>

- START_LINE – indicates, whether it is a **request or response**

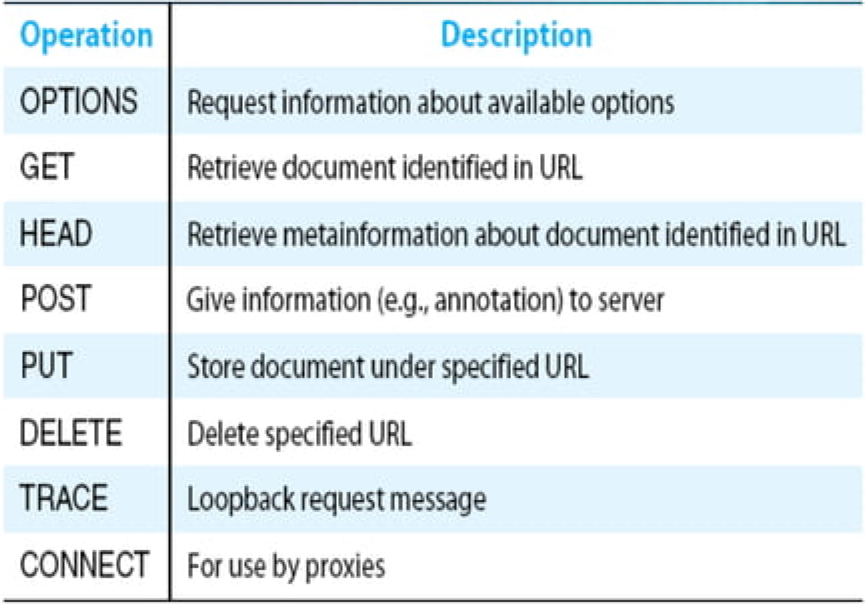
Request Messages

- The first line of an **HTTP request message specifies three things**: the operation to be performed, the Web page the operation should be performed on, and the version of HTTP being used.
- START_LINE - **GET** http://www.xyz.com/index.html **HTTP/1.1**

HyperText Transfer Protocol (HTTP)

START_LINE - **GET** index.html **HTTP/1.1**

MESSAGE_HEADER – **host:** www.xyz.com



Operation	Description
OPTIONS	Request information about available options
GET	Retrieve document identified in URL
HEAD	Retrieve metainformation about document identified in URL
POST	Give information (e.g., annotation) to server
PUT	Store document under specified URL
DELETE	Delete specified URL
TRACE	Loopback request message
CONNECT	For use by proxies

HyperText Transfer Protocol (HTTP)

Response Messages

- Response messages begin with a single START LINE and that line specifies the version of HTTP being used, a three-digit code indicating whether or not the request was successful, and a text string giving the reason for the response.

HTTP/1.1 202 Accepted

or

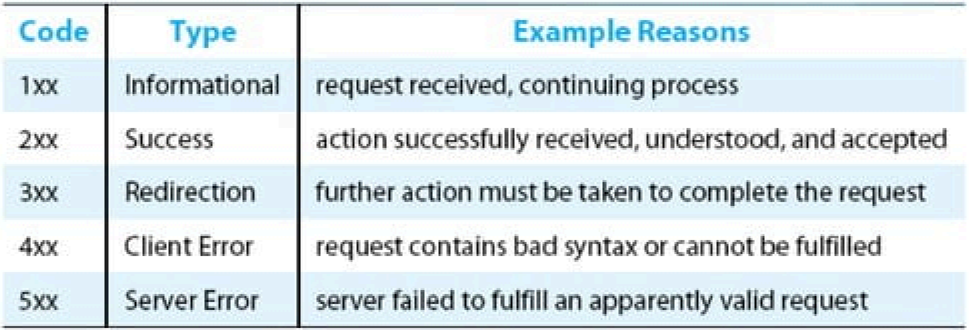
HTTP/1.1 404 Not Found

or

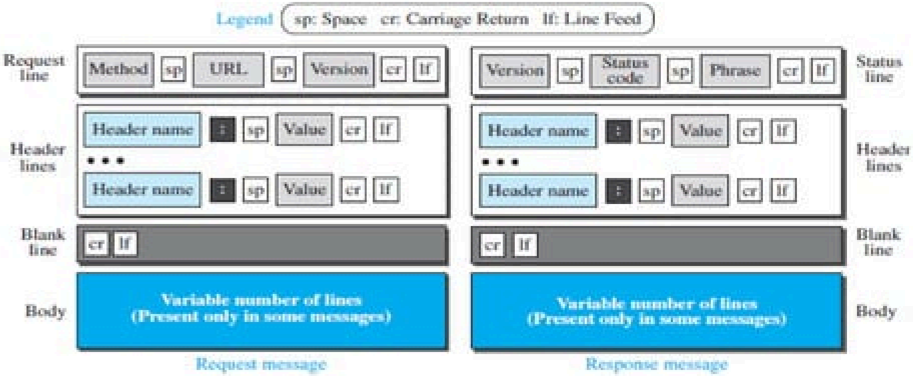
HTTP/1.1 301 Moved Permanently

Message Header - **Location : http://www.xyz1.com/index.html**

HyperText Transfer Protocol (HTTP)



Code	Type	Example Reasons
1xx	Informational	request received, continuing process
2xx	Success	action successfully received, understood, and accepted
3xx	Redirection	further action must be taken to complete the request
4xx	Client Error	request contains bad syntax or cannot be fulfilled
5xx	Server Error	server failed to fulfill an apparently valid request



HyperText Transfer Protocol (HTTP)

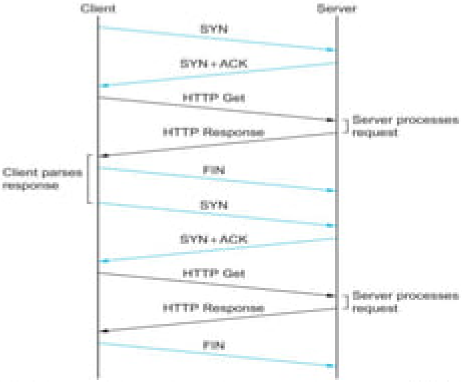
TCP Connections

- The **original version of HTTP (1.0) established a separate TCP connection for each data item retrieved** from the server.
- But it is **inefficient**: connection setup and teardown messages had to be exchanged between the client and server even if all the client wanted to do was verify that it had the most recent copy of a page.
- Thus, retrieving a page that included some text and a dozen icons or other small graphics would result in 13 separate TCP connections being established and closed.
- **HTTP 1.1,**
  - **Persistent Connection:** The client and server can exchange multiple request/response messages over the same TCP connection.

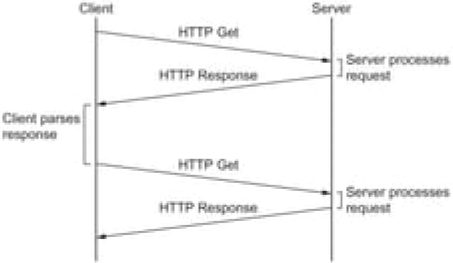
HyperText Transfer Protocol (HTTP)

Advantages of Persistent Connection

- **Eliminate the connection setup** overhead.
  - Thereby reducing the load on the server, the load on the network caused by the additional TCP packets, and the delay perceived by the user.
- A client can **send multiple request messages** down a single TCP connection.
  - TCP's congestion window mechanism is able to operate more efficiently. This is because it's not necessary to go through the slow start phase for each page.



HTTP 1.0 Behavior



HTTP 1.1 Behavior with persistent Connection

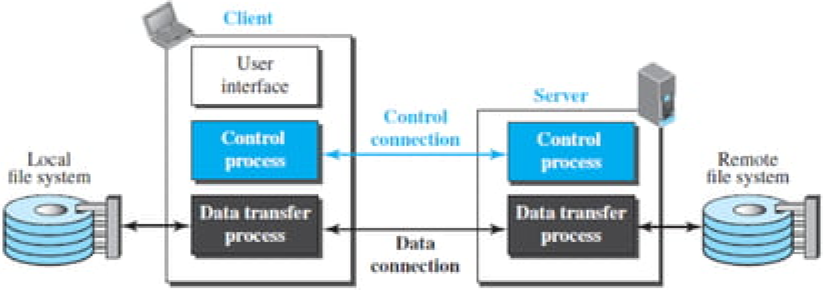
HyperText Transfer Protocol (HTTP)

Caching

- **Benefits of caching**
  - Client: Page can be retrieved from a nearby cache can be displayed much more quickly than if it has to be fetched from across the world.
  - Server: Reduces the load on the server.
- **Caching can be implemented on,**
  - **User's Browser:** Cache recently accessed pages, and simply display the cached copy if the user visits the same page again.
  - **Single site-wide cache:** The users within the site most likely know what machine is caching pages on behalf of the site, and they configure their browsers to connect directly to the caching host. This node is sometimes called a **proxy.**
  - **Middle of the Internet:** ISP can cache the pages.

File Transfer Protocol (FTP)

- **FTP** is the standard protocol provided by **TCP/IP** for **copying a file from one host to another**.



- The client has three components: the user interface, the client control process, and the client data transfer process.
- The server has two components: the server control process and the server data transfer process.
- The control connection is made between the control processes.
- The data connection is made between the data transfer processes.

File Transfer Protocol (FTP)

- Separation of commands and data transfer makes FTP more efficient.
- The control connection uses very simple rules of communication. (i.e.) only a line of command or a line of response at a time.
- The data connection needs more complex rules due to the variety of data types transferred.

Two Connections

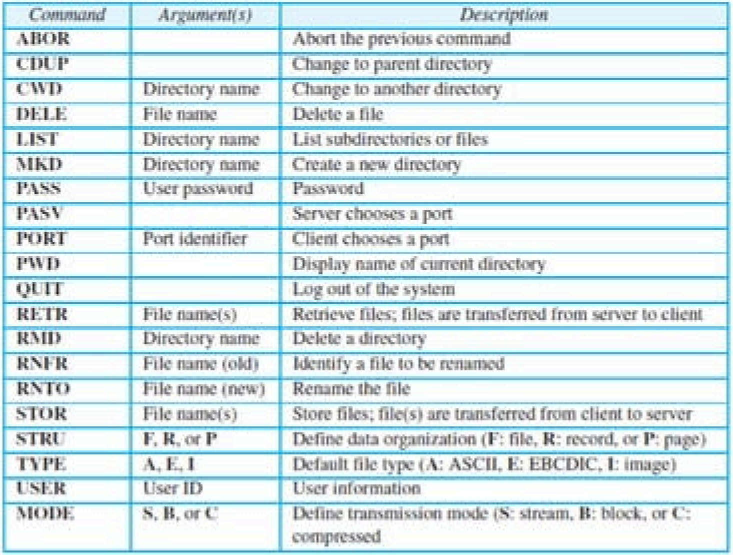
- When a user starts an FTP session, the control connection opens.
- While the control connection is open, the data connection can be opened and closed multiple times if several files are transferred.
- FTP uses two well-known TCP ports: **port 21** is used for the **control connection**, and **port 20** is used for the **data connection**.

File Transfer Protocol (FTP)

Control Connection

- Communication is achieved through commands and responses.
- Commands are sent from the client to the server and responses are sent from the server to the client.

Table 26.4 *Some FTP commands*



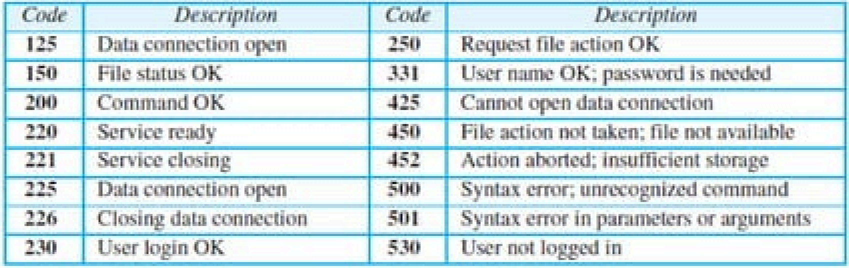
Command	*Argument(s)*	*Description*
ABOR		Abort the previous command
CDUP		Change to parent directory
CWD	Directory name	Change to another directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
MKD	Directory name	Create a new directory
PASS	User password	Password
PASV		Server chooses a port
PORT	Port identifier	Client chooses a port
PWD		Display name of current directory
QUIT		Log out of the system
RETR	File name(s)	Retrieve files; files are transferred from server to client
RMD	Directory name	Delete a directory
RNFR	File name (old)	Identify a file to be renamed
RNTO	File name (new)	Rename the file
STOR	File name(s)	Store files; file(s) are transferred from client to server
STRU	**F**, **R**, or **P**	Define data organization (**F**: file, **R**: record, or **P**: page)
TYPE	**A**, **E**, **I**	Default file type (**A**: ASCII, **E**: EBCDIC, **I**: image)
USER	User ID	User information
MODE	**S**, **B**, or **C**	Define transmission mode (**S**: stream, **B**: block, or **C**: compressed

File Transfer Protocol (FTP)

Control Connection

- Every FTP command generates at least one response.
- A response has two parts: a three-digit number followed by text.

Table 26.5 *Some responses in FTP*



Code	*Description*	*Code*	*Description*
125	Data connection open	**250**	Request file action OK
150	File status OK	**331**	User name OK; password is needed
200	Command OK	**425**	Cannot open data connection
220	Service ready	**450**	File action not taken; file not available
221	Service closing	**452**	Action aborted; insufficient storage
225	Data connection open	**500**	Syntax error; unrecognized command
226	Closing data connection	**501**	Syntax error in parameters or arguments
230	User login OK	**530**	User not logged in

File Transfer Protocol (FTP)

Data Connection

- The data connection uses the **well-known port 20** at the server site.
- The creation of a data connection is different from the control connection.
- The following shows the steps:

1. The **client**, not the server, issues a **passive open** using **an ephemeral port**. This must be done by the client because it is the client that issues the commands for transferring files.
2. Using the PORT command the client sends this port number to the server.
3. The **server receives the port number** and **issues an active open using the well-known port 20** and the received ephemeral port number.

File Transfer Protocol (FTP)

Data Connection – Communication over Data Connection

- The client must define the type of file to be transferred, the structure of the data, and the transmission mode.
- **File Type:** ASCII file, EBCDIC file, or image file.
- **Data Structure:** FTP can transfer a file across the data connection using one of the following interpretations of the structure of the data: **file structure, record structure, or page structure**.
- **Transmission Mode:** FTP can transfer a file across the data connection using one of the following three transmission modes: **stream mode, block mode, or compressed mode**.
- **File Transfer:** File transfer in FTP means one of three things: **retrieving a file** (server to client), **storing a file** (client to server), and **directory listing** (server to client).
- **Security in FTP:** One can add a Secure Socket Layer between the FTP application layer and the TCP layer.

Electronic Mail (E-Mail)

- Electronic mail (or e-mail) allows users to exchange messages.
- Client / Server model.
- To understand E-Mail,
  - Distinguish the user interface (Mail reader) from the underlying message transfer protocol (SMTP).
  - Distinguish between the transfer protocol and companion protocol (MIME).
- **Protocols Used**
  - **SMTP** (Simple Mail Transfer Protocol) – Used for message transfer.
  - **IMAP** (Internet Message access Protocol) – Used to retrieve message.
  - **POP** (Post office Protocol) – Used to retrieve Message.
  - **MIME** (Multipurpose Internet Mail Extensions) – Used to define the format of the message being exchanged.

Electronic Mail (E-Mail)

Multipurpose Internet Mail Extension (MIME)

Message Format

- RFC 822 defines two parts: **Header and Body**.

Header

- Series of <CRLF> terminated lines.
- The header is separated from the message by a blank line.
- Each header line contains,
  - **Type:** To, From, MIME - Version, Content - Type, Subject, Date
  - **Value**
- Type and value is separated by a colon.
- E-mail message can have many different types of data (like images, videos, documents, audio, etc…).
- These different types of data are defined in MIME.

Electronic Mail (E-Mail)

Multipurpose Internet Mail Extension (MIME) – contains three parts.

- **First piece of information:** Collection of header lines that augments the original set defined by RFC822.
  - It describes the data being carried in the message.
  - It contains **MIME-Version, Content-Type, From, To, Content-Description (Subject), Date**.
- **Second piece of information:** Defines the content type of the message. i.e., **"Content-Type"** field.
  - **Examples:** image/jpeg, image/gif

    text/plain, text/richtext

    audio/basic, multipart/mixed

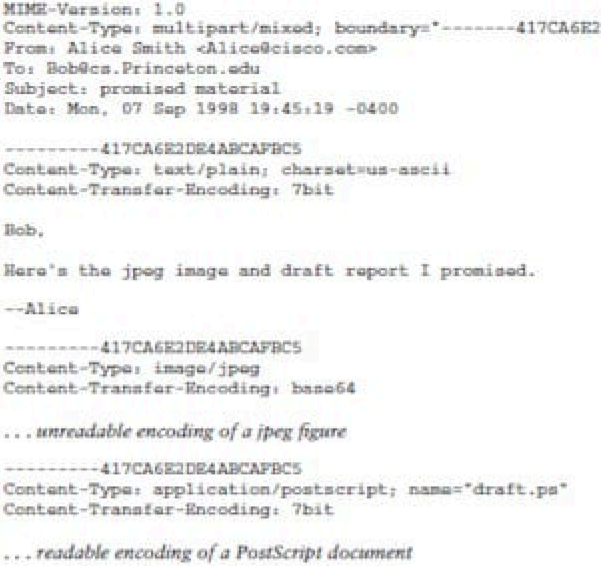
    video/mpeg, video/quicktime

    application/pdf, application/zip, application/postscript

Electronic Mail (E-Mail)

Multipurpose Internet Mail Extension (MIME) – contains three parts.

- **Third piece of information:** Defines the content encoding scheme for the message. i.e., **"Content-Transfer-Encoding"** field.
  - **Example:** 7 bit, base 64, quoted-printable.



```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="-------417CA6E2DE4ABCAFBC5"
From: Alice Smith <Alice@cisco.com>
To: Bob@cs.Princeton.edu
Subject: promised material
Date: Mon, 07 Sep 1998 19:45:19 -0400

---------417CA6E2DE4ABCAFBC5
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Bob,

Here's the jpeg image and draft report I promised.

--Alice

---------417CA6E2DE4ABCAFBC5
Content-Type: image/jpeg
Content-Transfer-Encoding: base64

. . . unreadable encoding of a jpeg figure

---------417CA6E2DE4ABCAFBC5
Content-Type: application/postscript; name="draft.ps"
Content-Transfer-Encoding: 7bit

. . . readable encoding of a PostScript document
```

Electronic Mail (E-Mail)

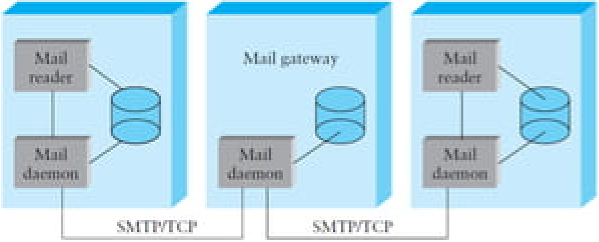
Simple Mail Transfer Protocol (SMTP)

- SMTP is used to transfer message from one host to another host.
- **Key players:**
  - **Mail reader:** Web browser
  - **Mail daemon:** Process runs on host
- **Mail Daemon**
  - Act as a **Mail Transfer Agent (MTA)**.
  - The daemon uses SMTP running over TCP to transmit the message to a daemon running on another machine, and the daemon puts incoming messages into the user's mailbox.
  - MTA on a sender's machine establishes an SMTP/TCP connection to the MTA on the recipient's mail server, in many cases the mail traverses one or more mail gateways on its route from the sender's host to the receiver's host.

Electronic Mail (E-Mail)

Simple Mail Transfer Protocol (SMTP)

- **Mail Daemon**
  - Job of the gateway is to store and forward email messages, much like an "**IP gateway**".
  - But the difference is, mail gateway typically buffers messages on disk and is willing to try retransmitting them to the next machine for several days, while an IP router buffers datagrams in memory and is only willing to retry transmitting them for a fraction of a second.



- Some of the **commands used by the client** are HELO, MAIL, RCPT, DATA, QUIT.
- Server **response with code**.

Electronic Mail (E-Mail)

Mail Reader

- Users actually retrieve his or her messages from the mailbox, read them, reply to them, and possibly save a copy for future reference.
- The user performs all these actions by interacting with a mail reader.
- The reader was a program running on the same machine as the user's mailbox, in which case it could simply read and write the file that implements the mailbox.
- Protocols used to retrieve mails are,
  - **POP** (Post Office Protocol)
  - **IMAP** (Internet Message Access Protocol)

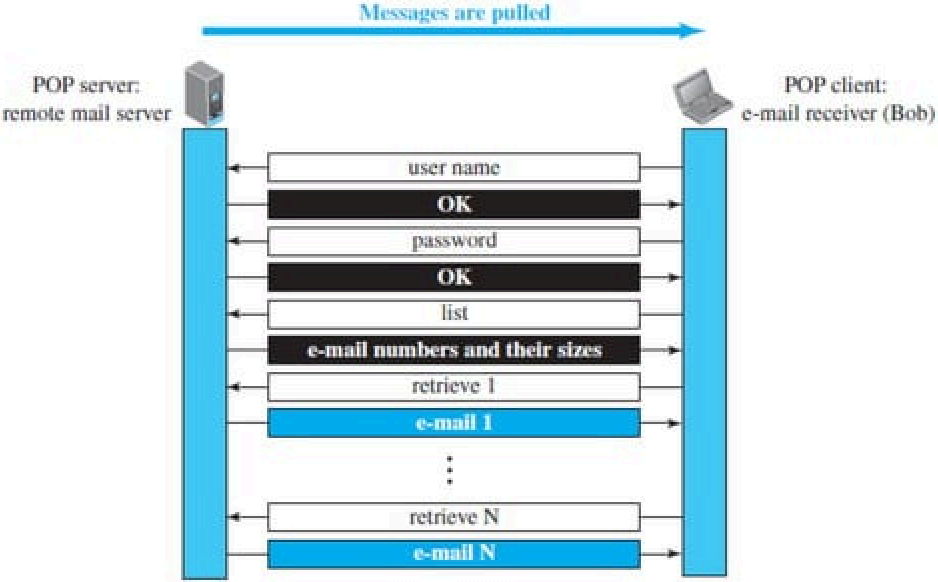
Electronic Mail (E-Mail)

Post Office Protocol (POP3)

- It is simple but limited in functionality.
- The **client POP3 software** is installed on the **recipient computer**.
- The **server POP3 software** is installed on the **mail server**.
- Mail access starts with the client when the user needs to download its e-mail from the mailbox on the mail server.
- The client opens a connection to the server on TCP **port 110**.
- It then sends its user name and password to access the mailbox.
- The user can then list and retrieve the mail messages, one by one.
- POP3 has two modes: the ***delete* mode** and the ***keep* mode**.
- In the **delete mode**, the **mail is deleted** from the mailbox after each retrieval.
- In the **keep mode**, the **mail remains in the mailbox after retrieval**.

Electronic Mail (E-Mail)

Post Office Protocol (POP3)



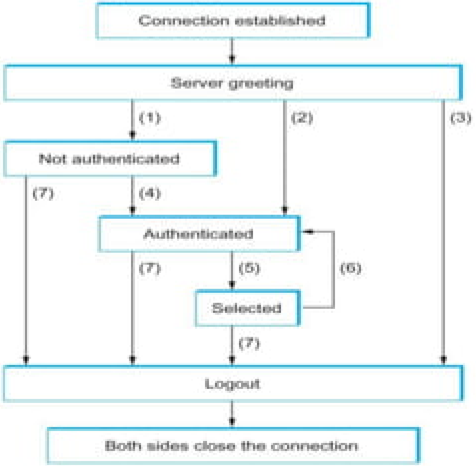
Electronic Mail (E-Mail)

Internet Message Access Protocol (IMAP)

- It is a client/server protocol running over TCP
- The client (running on the user's desktop machine) issues commands in the form of <CRLF> - terminated ASCII text lines and the mail server (running on the machine that maintains the user's mailbox) responds in-kind.
- The exchange begins with the client authenticating him or herself, and identifying the mailbox he or she wants to access.
- Commands used by IMAP are:
  - LOGIN
  - AUTHENDICATE
  - SELECT
  - EXAMINE
  - CLOSE & LOGOUT

Electronic Mail (E-Mail)

Internet Message Access Protocol (IMAP)



Connection established

Server greeting

Not authenticated

Authenticated

Selected

Logout

Both sides close the connection

(1) Connection without preauthentication (OK greeting)
(2) Preauthenticated connection (PREAUTH greeting)
(3) Rejected connection (BYE greeting)
(4) Successful LOGIN or AUTHENTICATE command
(5) Successful SELECT or EXAMINE command
(6) CLOSE command, or failed SELECT or EXAMINE command
(7) LOGOUT command, server shutdown, or connection closed

E-Mail Security

- To provide E-Mail security, Pretty Good Privacy (PGP) and Secure/Multipurpose Internet Mail Extensions (S/MIME) are used.

TErminaL NETwork (TELNET)

- One of the original **remote logging protocols** is **TELNET**.
- Although **TELNET requires a logging name and password**, it is vulnerable to hacking because it sends all data including the password in plaintext.
- A **hacker can eavesdrop** and obtain the logging name and password.
- TELNET allows us to **explain the issues and challenges** related to the concept of **remote logging**, which is also used in SSH when it serves as a remote logging protocol.
- **Network administrators** often use TELNET for **diagnostic and debugging purposes**.

Local Logging

- When a user logs into a local system, it is called **local logging**.
- A user types at a terminal or at a workstation running a terminal emulator, the keystrokes are accepted by the terminal driver.
- The terminal driver passes the characters to the operating system.
- The operating system interprets the combination of characters and invokes the desired application program or utility.

TErminaL NETwork (TELNET)

Remote Logging

- When a **user wants to access an application program** or utility located **on a remote machine**, she performs **remote logging**.
- Here the **TELNET client and server programs** come into use.
- The user sends the keystrokes to the terminal driver where the local operating system accepts the characters but does not interpret them.
- The characters are sent to the TELNET client, which transforms the characters into a universal character set called **Network Virtual Terminal (NVT)** characters and delivers them to the local TCP/IP stack.
- The commands or text, in NVT form, travel through the Internet and arrive at the TCP/IP stack at the remote machine.

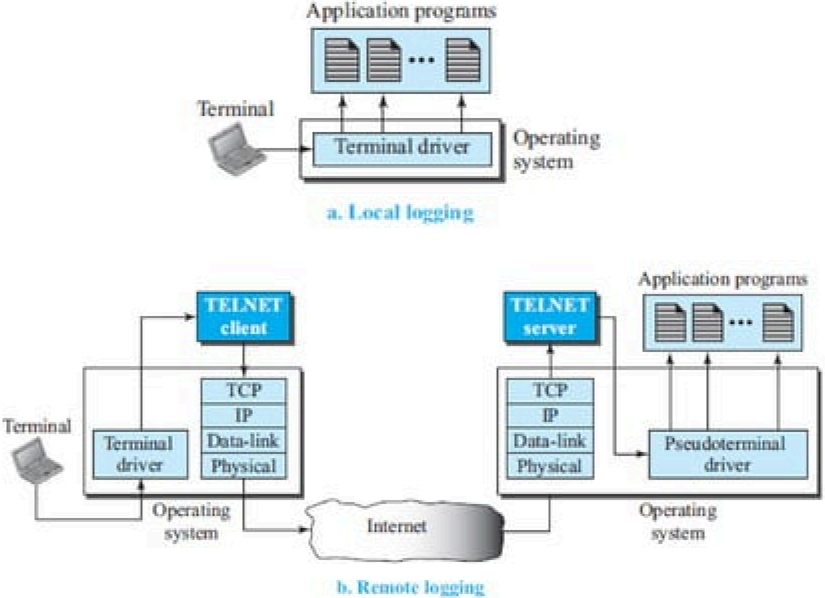
TErminaL NETwork (TELNET)

Remote Logging

- Here the characters are delivered to the operating system and passed to the TELNET server, which changes the characters to the corresponding characters understandable by the remote computer.
- However, the characters cannot be passed directly to the operating system because the remote operating system is not designed to receive characters from a TELNET server; it is designed to receive characters from a terminal driver.
- The solution is to add a piece of software called a **pseudoterminal driver**, which pretends that the characters are coming from a terminal.
- The operating system then passes the characters to the appropriate application program.

TErminaL NETwork (TELNET)

Local versus Remote Logging

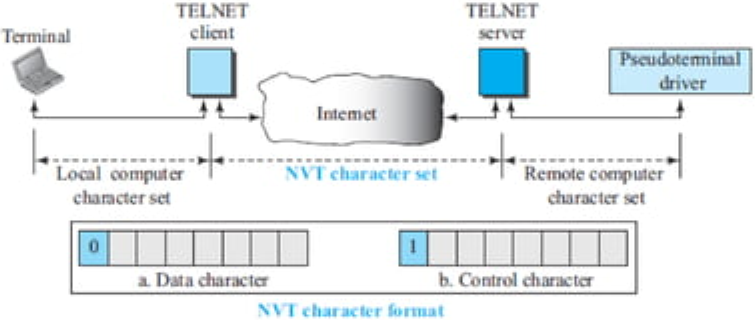


a. Local logging

b. Remote logging

TErminaL NETwork (TELNET)

- TELNET solves heterogeneity by defining a universal interface called the **Network Virtual Terminal (NVT)** character set.
- Via this interface, the client TELNET translates characters (data or commands) that come from the local terminal into NVT form and delivers them to the network.
- The server TELNET, translates data and commands from NVT form into the form acceptable by the remote computer.
- **NVT uses two sets of characters,** one for **data** and one for **control**.

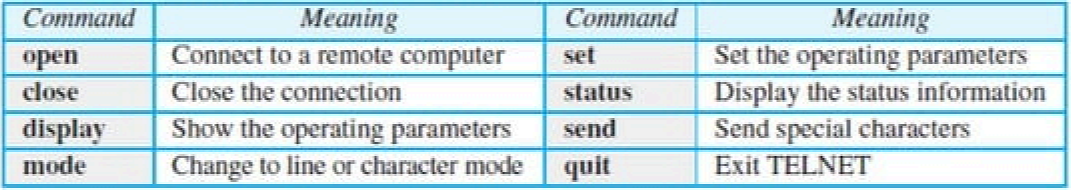


TErminaL NETwork (TELNET)

User Interface

- The operating system (UNIX, for example) defines an interface with user-friendly commands.

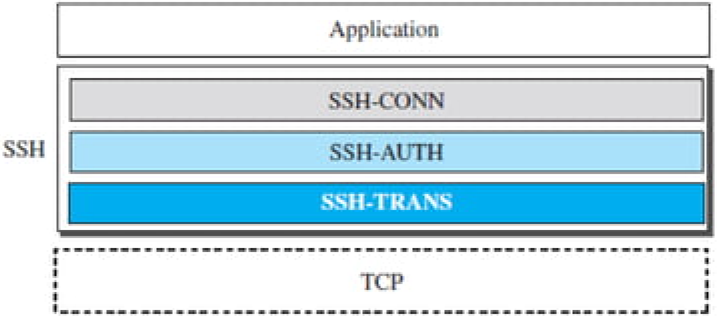
Table 26.11 *Examples of interface commands*



Command	*Meaning*	*Command*	*Meaning*
open	Connect to a remote computer	**set**	Set the operating parameters
close	Close the connection	**status**	Display the status information
display	Show the operating parameters	**send**	Send special characters
mode	Change to line or character mode	**quit**	Exit TELNET

Secure Shell (SSH)

- **Secure Shell (SSH)** is a secure application program that can be used for remote logging and file transfer.
- **Two versions of SSH:** SSH-1 and SSH-2, which are totally incompatible.
- The first version, **SSH-1**, is now **deprecated because of security flaws** in it.
- **Components:** SSH is an application-layer protocol with **three components**.



Secure Shell (SSH)

SSH Transport-Layer Protocol (SSH-TRANS)

- Since **TCP is not a secured transport-layer protocol**, SSH uses a protocol that creates a secured channel on top of the TCP. It is referred to as **SSH-TRANS**.
- The client and server first use the **TCP protocol to establish an insecure connection**.
- Then they **exchange several security parameters to establish a secure channel** on top of the TCP.
- **List of services provided by this protocol:**

1. **Privacy or confidentiality** of the message exchanged.
2. **Data integrity**, which means that it is guaranteed that the messages exchanged between the client and server are not changed by an intruder.
3. **Server authentication**, which means that the client is now sure that the server is the one that it claims to be.
4. **Compression of the messages**, which improves the efficiency of the system and makes attack more difficult.

Secure Shell (SSH)

SSH Authentication Protocol (SSH-AUTH)

- After a secure channel is established between the client and the server and the server is authenticated for the client, SSH can call another procedure that can authenticate the client for the server.
- Authentication starts with the client, which **sends a request message to the server**.
- The request includes the **user name, server name, the method of authentication, and the required data**.
- The **server responds with either a success message**, which confirms that the client is authenticated, **or a failed message**, which means that the process needs to be repeated with a new request message.

Secure Shell (SSH)

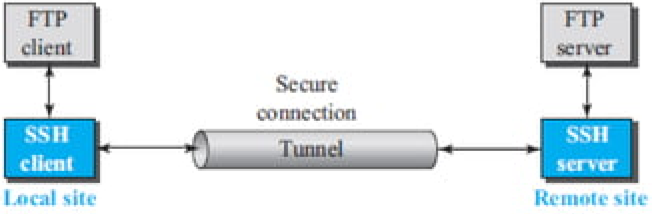
SSH Connection Protocol (SSH-CONN)

- After the secured channel is established and both server and client are authenticated for each other, **SSH can call a piece of software that implements the third protocol, SSHCONN**.
- One of the **services provided** by the **SSH-CONN protocol is multiplexing**.
- SSH-CONN takes the secure channel established by the two previous protocols and lets the **client create multiple logical channels** over it.
- Each channel can be used for a different purpose, such as remote logging, file transfer, and so on.

Secure Shell (SSH)

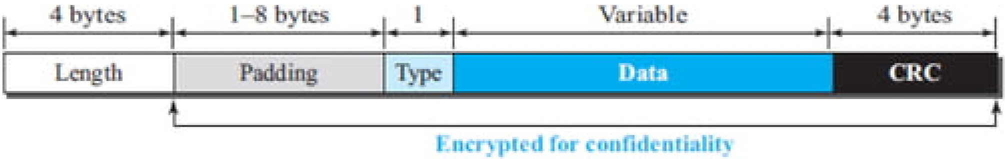
Applications

- **SSH for Remote Logging:** Commercial applications use SSH for remote logging. Eg: PuTTy and Tectia
- **SSH for File Transfer:** The application program uses one of the channels provided by the SSH to transfer files. Eg: Secure FTP, Secure Copy
- **Port Forwarding:** The SSH port forwarding mechanism creates a tunnel through which the messages belonging to other protocols can travel.



Secure Shell (SSH)

Format of the SSH Packets



- **Length:** defines the length of the packet but does not include the padding.
- **Padding:** Added to the packet to make the attack on the security provision more difficult.
- **Type:** Defines the type of the packet used in different SSH protocols.
- **Data:** The data transferred by the packet in different protocols.
- **Cyclic redundancy check (CRC):** Used for error detection.

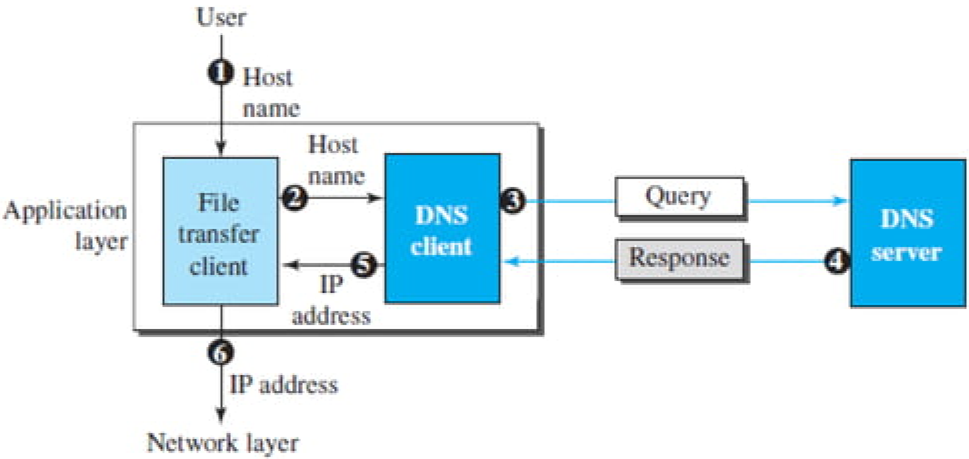
Domain Name System (DNS)

- TCP/IP uses a DNS client and a DNS server **to map a name to an address**.
- The following **six steps map the host name to an IP address:**

1. The user passes the host name to the file transfer client.
2. The file transfer client passes the host name to the DNS client.
3. Each computer, after being booted, knows the address of one DNS server. The DNS client sends a message to a DNS server with a query that gives the file transfer server name using the known IP address of the DNS server.
4. The DNS server responds with the IP address of the desired file transfer server.
5. The DNS server passes the IP address to the file transfer client.
6. The file transfer client now uses the received IP address to access the file transfer server.

Domain Name System (DNS)

Six steps to map the host name to an IP address



Domain Name System (DNS)

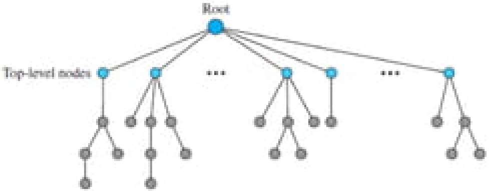
Name Space

- **Maps each address to a unique name** can be organized in two ways: **flat or hierarchical**.
- **Flat name space:** A name is assigned to an address. A name is a sequence of characters without structure.
- **Hierarchical name space:** Each name is made of several parts. The first part can define the nature of the organization, the second part can define the name of an organization, the third part can define departments in the organization, and so on.

Domain Name System (DNS)

Domain Name Space

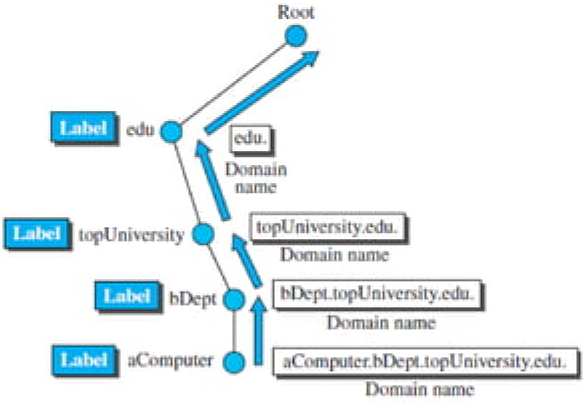
- To have a hierarchical name space, a **domain name space** was designed.
- In this design the names are defined in an inverted-tree structure with the root at the top.
- **Label:** Each node in the tree has a **label**, which is a string with a maximum of 63 characters. The root label is a null string (empty string).
- **Domain Name:** Each node in the tree has a domain name. A full **domain name** is a sequence of labels separated by dots (.). The domain names are always read from the node up to the root. The last label is the label of the root (null).
- If a label is terminated by a null string, it is called a **fully qualified domain name (FQDN)**.
- If a label is not terminated by a null string, it is called a **partially qualified domain name (PQDN)**.



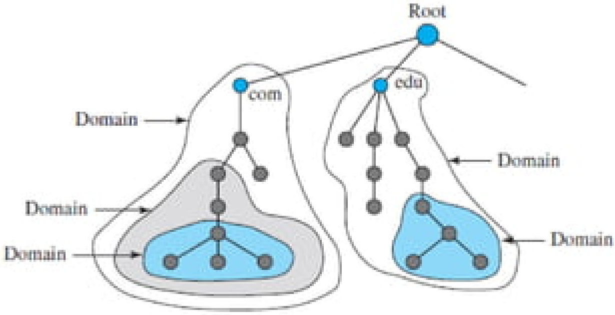
Domain Name System (DNS)

Domain

- A **domain** is a subtree of the domain name space.
- The name of the domain is the name of the node at the top of the subtree.



Domain names and labels



Domains

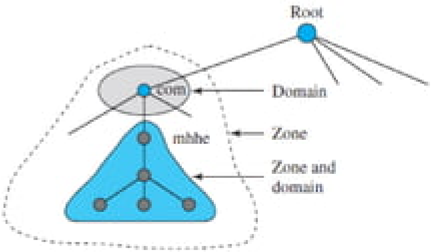
Domain Name System (DNS)

Distribution of Name Space

- The information contained in the domain name space must be stored.
- **Hierarchy of Name Servers:** The solution is to distribute the information among many computers called **DNS servers.**

Zone

- Since the complete domain name hierarchy cannot be stored on a single server, it is divided among many servers.
- What a server is responsible for or has authority over is called a **zone**.



Domain Name System (DNS)

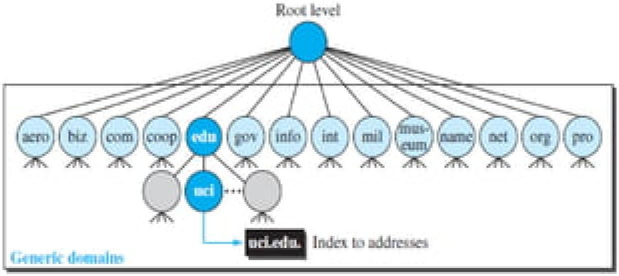
Root Server

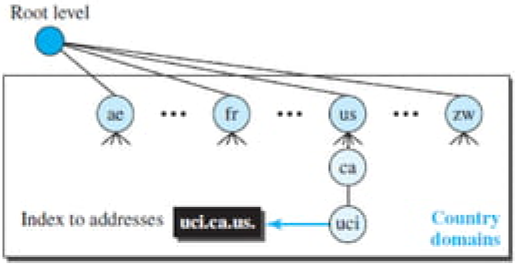
- A **root server** is a server whose zone consists of the whole tree.
- A root server usually **does not store any information about domains** but delegates its authority to other servers, **keeping references to those servers**.
- DNS defines two types of servers: primary and secondary.
- **Primary server:** It stores a file about the zone for which it is an authority. It is responsible for creating, maintaining, and updating the zone file. It stores the zone file on a local disk.
- **Secondary server:** It transfers the complete information about a zone from another server (primary or secondary) and stores the file on its local disk. The secondary server neither creates nor updates the zone files.

Domain Name System (DNS)

DNS in Internet

- DNS is a protocol that can be used in different platforms.
- In the Internet, the domain name space (tree) was originally divided into three different sections: generic domains, country domains, and the inverse domains.
- **Generic Domain:** Define registered hosts according to their generic behavior.
- **Country Domain:** Section uses two-character country abbreviations.





Domain Name System (DNS)

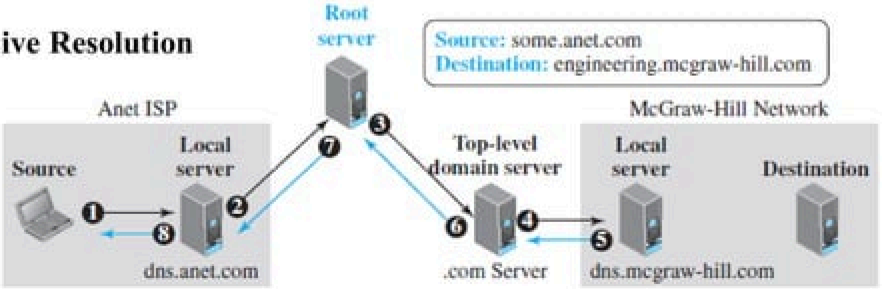
Resolution

- Mapping a name to an address is called **name-address resolution**.
- **DNS** is designed as **a client-server application**.
- A host that needs to map an address to a name or a name to an address calls a DNS client called a **resolver**.
- The **resolver accesses the closest DNS server** with a mapping request.
- If the server has the information, it satisfies the resolver; otherwise, it either refers the resolver to other servers or asks other servers to provide the information.
- After **the resolver receives the mapping**, it **interprets the response** to see if it is a real resolution or an error, and finally delivers the result to the process that requested it.
- A resolution can be either **recursive** or **iterative**.

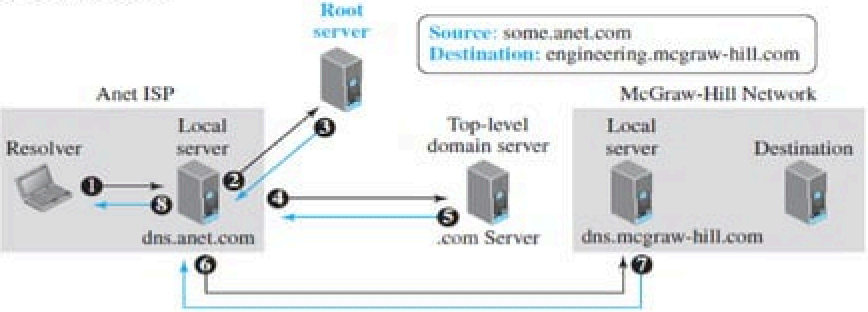
Domain Name System (DNS)

Resolution

Recursive Resolution



Iterative Resolution



Domain Name System (DNS)

Caching

- Each time **a server receives a query for a name that is not in its domain**, it needs to search its database for a server IP address.
- **Reduction of this search time** would increase efficiency.
- DNS handles this with a mechanism called **caching**.

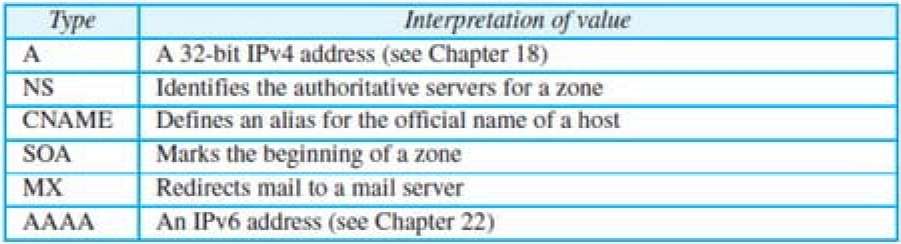
Resource Records

- The **zone information** associated with a server is implemented as a set of **resource records**.
- A resource record is a 5-tuple structure: **(Domain Name, Type, Class, TTL, Value)**.

Domain Name System (DNS)

Resource Records

- **Domain Name:** Identifies the resource record
- **Class:** Defines the type of network
- **Value:** Information kept about the domain name
- **TTL:** The number of seconds for which the information is valid.
- **Type:** Defines how the value should be interpreted.

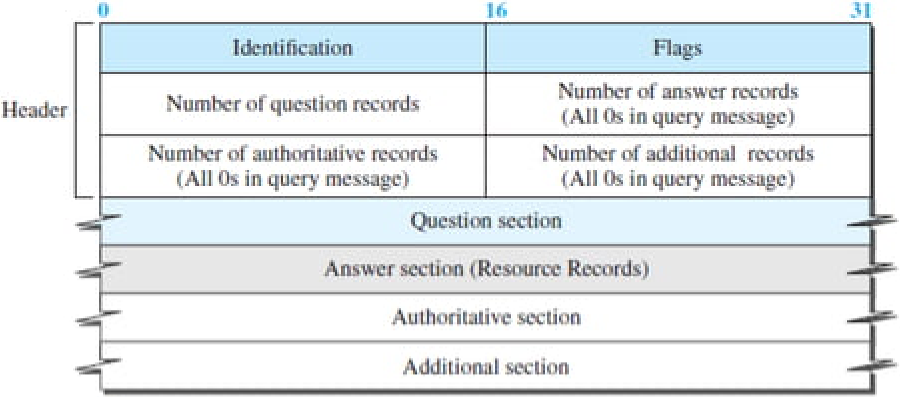


Type	*Interpretation of value*
A	A 32-bit IPv4 address (see Chapter 18)
NS	Identifies the authoritative servers for a zone
CNAME	Defines an alias for the official name of a host
SOA	Marks the beginning of a zone
MX	Redirects mail to a mail server
AAAA	An IPv6 address (see Chapter 22)

Domain Name System (DNS)

DNS Message

- To retrieve information about hosts, **DNS uses two types of messages: query and response**.


0
16
31
Header
Identification
Flags
Number of question records
Number of answer records
(All 0s in query message)
Number of authoritative records
(All 0s in query message)
Number of additional records
(All 0s in query message)
Question section
Answer section (Resource Records)
Authoritative section
Additional section

Note:
The query message contains only the question section.
The response message includes the question section,
the answer section, and possibly two other sections.

Domain Name System (DNS)

DNS Message

- **Identification field:** Used by the client to match the response with the query.
- **Flag field:** Defines whether the message is a query or response. It also includes status of error.
- **Next four fields:** In the header define the number of each record type in the message.
- **Question section:** Consists of one or more question records. It is present in both query and response messages.
- **Answer section:** Consists of one or more resource records. It is present only in response messages.
- **Authoritative section:** Gives information (domain name) about one or more authoritative servers for the query.
- **Additional information section:** Provides additional information that may help the resolver.

Domain Name System (DNS)

Encapsulation

- **DNS** can **use either UDP or TCP**.
- In both cases the well-known port used by the server is **port 53**.
- **UDP** is used **when the size of the response message is less than 512 bytes** because most UDP packages have a 512-byte packet size limit.
- If the **size of the response message is more than 512 bytes**, a **TCP** connection is used.

Registrars

- **How are new domains added to DNS?**
- This is done through a **registrar**, a commercial entity accredited by ICANN.
- A **registrar first verifies that the requested domain name is unique** and then enters it into the DNS database.
- A **fee** is charged.

Domain name: ws.wonderful.com **IP address:** 200.200.200.5

Domain Name System (DNS)

Dynamic Domain Name System (DDNS)

- When a binding between a name and an address is determined, **the information is sent, usually by DHCP to a primary DNS server**.
- The primary server updates the zone.
- The secondary servers are notified either actively or passively.
- To provide security and prevent unauthorized changes in the DNS records, DDNS can use an authentication mechanism.

Security in DNS

- **DNS Security (DNSSEC)** that provides message origin **authentication and message integrity** using a security service called **digital signature**.

Domain Name System (DNS)

Dynamic Domain Name System (DDNS)

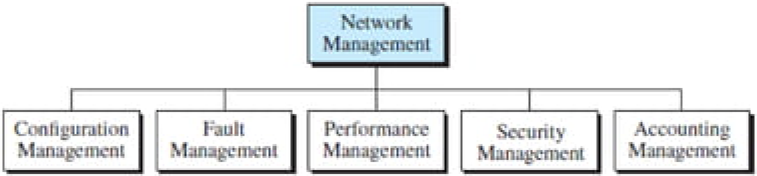
- When a binding between a name and an address is determined, **the information is sent, usually by DHCP to a primary DNS server**.
- The primary server updates the zone.
- The secondary servers are notified either actively or passively.
- To provide security and prevent unauthorized changes in the DNS records, DDNS can use an authentication mechanism.

Security in DNS

- **DNS Security (DNSSEC)** that provides message origin **authentication and message integrity** using a security service called **digital signature**.

Simple Network Management Protocol (SNMP)

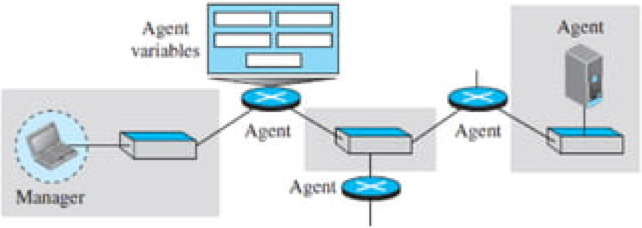
- Used for monitoring, testing, configuring, and troubleshooting network components to meet a set of requirements defined by an organization.
- **Areas of Network Management**



- **SNMP** is a **framework** for managing devices in an internet using the TCP/IP protocol suite.
- It provides a set of fundamental operations for monitoring and maintaining an internet.
- SNMP uses the concept of manager and agent.
- A manager, usually a host, controls and monitors a set of agents, usually routers or servers.

Simple Network Management Protocol (SNMP)

- SNMP is an **application-level protocol** in which **a few manager stations control a set of agents**.
- It can **monitor devices** made by different manufacturers and installed on different physical networks.



- A management station, called a **manager**, is a host that runs the **SNMP client program**.
- A managed station, called an **agent**, is a router (or a host) that runs the **SNMP server program**.
- Management is achieved through simple interaction between a manager and an agent.

Simple Network Management Protocol (SNMP)

Management Components

- To do management tasks, SNMP uses two other protocols: **Structure of Management Information (SMI)** and **Management Information Base (MIB)**.

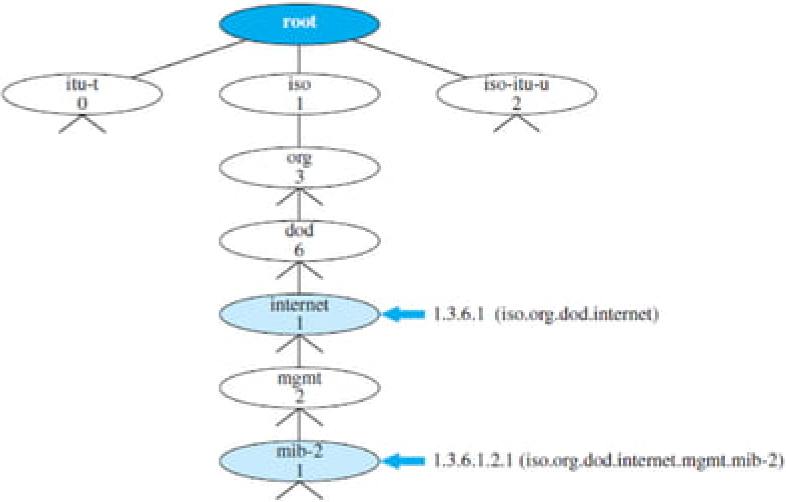
Structure of Management Information (SMI)

- Defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values.
- **Name:**
  - Each managed object have a unique name. To name objects globally, SMI uses an **object identifier,** which is a hierarchical identifier based on a tree structure.
  - The tree structure starts with an **unnamed root**.
  - **Each object** can be defined using **a sequence of integers separated by dots**.
  - The tree structure can also define an object using a sequence of textual names separated by dots.

Simple Network Management Protocol (SNMP)

Structure of Management Information (SMI)

- **Name:**
  - **Example:** iso.org.dod.internet.mgmt.mib-2 ↔ 1.3.6.1.2.1


root
itu-t
0
iso
1
iso-itu-u
2
org
3
dod
6
internet
1
mgmt
2
mib-2
1
1.3.6.1 (iso.org.dod.internet)
1.3.6.1.2.1 (iso.org.dod.internet.mgmt.mib-2)

Simple Network Management Protocol (SNMP)

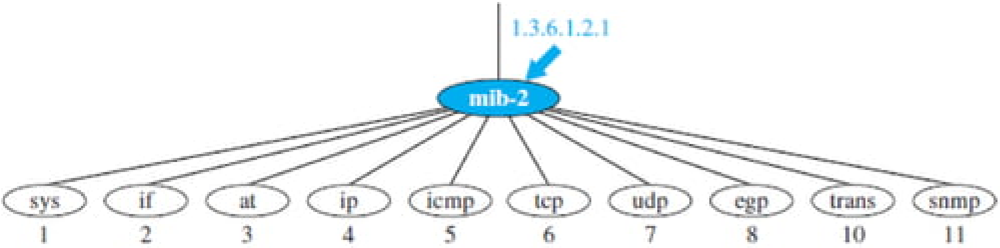
Structure of Management Information (SMI)

- **Type:**
  - The second attribute of an object is the type of data stored in it. To define the data type, SMI uses **Abstract Syntax Notation One (ASN.1)**.
  - SMI has two broad categories of data types: **simple** and **structured**.
  - **Simple data types:** Atomic data types.
  - **Structured data types:** By combining simple and structured data types.
- **Encoding Method:**
  - SMI uses **Basic Encoding Rules (BER),** to encode data to be transmitted over the network.
  - BER specifies that each piece of data be encoded in **triplet format**: **tag, length, and value (TLV)**.

Simple Network Management Protocol (SNMP)

Management Information Base (MIB)

- Creates a collection of named objects, their types, and their relationships to each other in an entity to be managed.
- The objects in MIB2 are categorized under several groups: system, interface, address translation, ip, icmp, tcp, udp, egp, transmission, and snmp.



Some MIB-2 groups

Simple Network Management Protocol (SNMP)

- SNMP uses both SMI and MIB in Internet network management.
- It is an application program that allows:
  - A manager to retrieve the value of an object defined in an agent.
  - A manager to store a value in an object defined in an agent.
  - An agent to send an alarm message about an abnormal situation to the manager.

SNMP PDUs

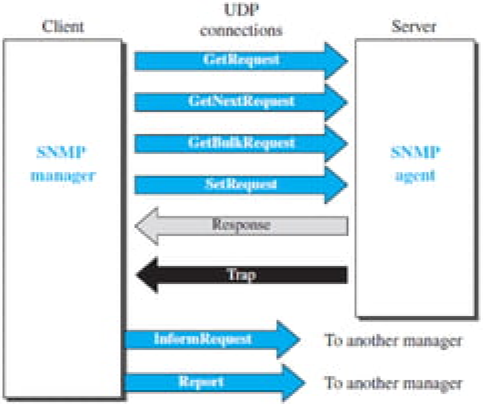
SNMPv3 defines **eight types of protocol data units** (or PDUs):

1. GetRequest, 2. GetNext- Request,

3. GetBulkRequest, 4. SetRequest,

5. Response, 6. Trap,

7. InformRequest, and 8. Report



Simple Network Management Protocol (SNMP)

SNMP PDUs

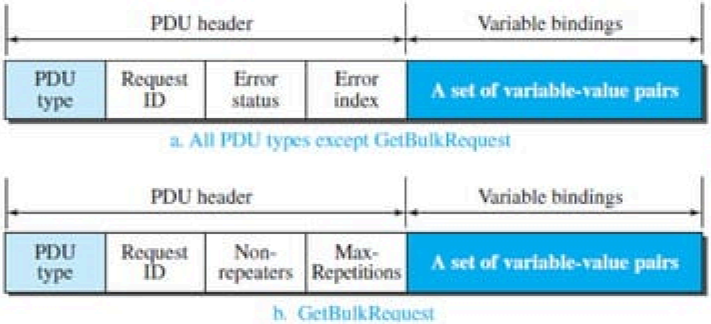
- **Get Request:** Manager to agent request **to retrieve the value of variable**. Agent will respond with requested variable with current stored value.
- **SetRequest:** Manager to agent request **to change value of variable**. Variable bindings are defined in request. Agent will respond with new value.
- **GetNextRequest:** Manager to agent request **to discover available variables and their values**. Agent will respond with value of next variable.
- **GetBulkRequest:** It **request multiple iterations** of GetNextRequest. Returns the response with multiple variable bindings in request.
- **Response: Returns value as requested from agent to manager**. It used as response to set and get request.

Simple Network Management Protocol (SNMP)

SNMP PDUs

- **Trap:** It is **an notification event**, agent to manager which is not requested by manager. **Agent itself informing to manager**.
- **Inform request:** It is **manger to manger communication**, one manager can send some information to another manager using informRequest PDU receiving manger response with Response PDU to manger confirming receipt of manager.

SNMP PDU Format



PDU header				Variable bindings
PDU type	Request ID	Error status	Error index	A set of variable-value pairs

a. All PDU types except GetBulkRequest

PDU header				Variable bindings
PDU type	Request ID	Non-repeaters	Max-Repetitions	A set of variable-value pairs

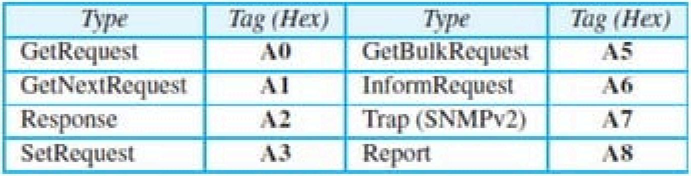
b. GetBulkRequest

Note:
The error status and error index values are set to 0 for all request messages.

Simple Network Management Protocol (SNMP)

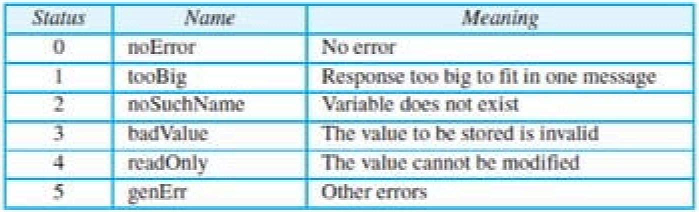
SNMP PDU Format

- **PDU type:** Defines the type of the PDU.



Type	*Tag (Hex)*	*Type*	*Tag (Hex)*
GetRequest	A0	GetBulkRequest	A5
GetNextRequest	A1	InformRequest	A6
Response	A2	Trap (SNMPv2)	A7
SetRequest	A3	Report	A8

- **Request ID: A sequence number used by the manager in a request PDU** and repeated by the agent in a response. It is used to match a request to a response.
- **Error status:** An **integer that is used only in response PDUs to show the types of errors** reported by the agent. Its value is 0 in request PDUs.



Status	*Name*	*Meaning*
0	noError	No error
1	tooBig	Response too big to fit in one message
2	noSuchName	Variable does not exist
3	badValue	The value to be stored is invalid
4	readOnly	The value cannot be modified
5	genErr	Other errors

Simple Network Management Protocol (SNMP)

SNMP PDU Format

- **Non-repeaters:** Used only in a GetBulkRequest PDU. It defines the number of non-repeating (regular objects) at the start of the variable value list.
- **Error index:** The error index is an offset that **tells the manager which variable caused the error**.
- **Max-repetitions:** Used only in a GetBulkRequest PDU. It defines **the maximum number of iterations** in the table to read all repeating objects.
- **Variable-value pair list:** A set of variables with the corresponding values the manager wants to retrieve or set. The values are null in request PDUs.

Simple Network Management Protocol (SNMP)

UDP Ports

- SNMP uses the **services of UDP** on **two well-known ports, 161** and **162**.
- The well-known port **161 is used by the server** (agent), and the well-known port **162 is used by the client** (manager).

Security

- SNMPv3 has added two new features like **security** and **remote administration**.
- Different aspects of security can be configured like **message authentication, confidentiality,** and **integrity**.