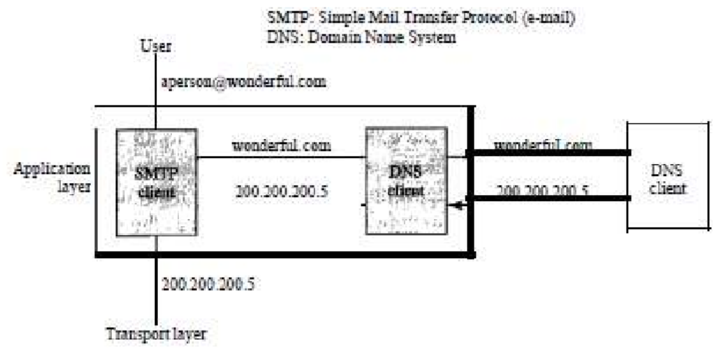
APPLICATION LAYER

1. DOMAIN NAME SYSTEM

The Domain Name System (DNS) is a supporting program that is used by other programs such as e-mail. Figure 25.1 shows an example of how a DNS client/server program can support an e-mail program to find the IP address of an e-mail recipient. A user of an e-mail program may know the e-mail address of the recipient; however, the IP protocol needs the IP address. The DNS client program sends a request to a DNS server to map the e-mail address to the corresponding IP address.



NAME SPACE: A name space that maps each address to a unique name can be organized in two ways: fiat or hierarchical.

Flat Name Space

In a flat name space, a name is assigned to an address. A name in this space is a sequence of characters without structure. The names mayor may not have a common section; if they do, it has no meaning. The main disadvantage of a fiat name space is that it cannot be used in a large System such as the Internet because it must be centrally controlled to avoid ambiguity and duplication.

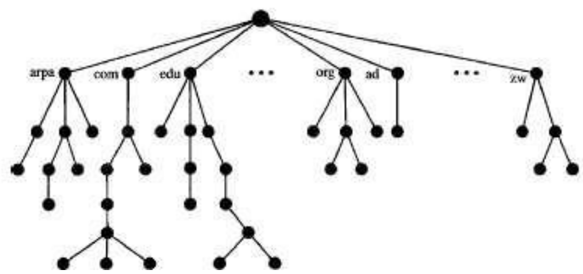
Hierarchical Name Space

In a 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 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. The tree can have only 128 levels: level 0 (root) to level 127 (see Figure 25.2).

Figure 25.2 *Domain name space*



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). DNS requires that children of a node (nodes that branch from the same node) have different labels, which guarantees the uniqueness of the domain names.

Domain Names

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). This means that a full domain name always ends in a null label, which means the last character is a dot because the null string is nothing. Figure 25.3 shows some domain names.

Fully Qualified Domain Name

If a label is terminated by a null string, it is called a fully qualified domain name (FQDN). An FQDN is a domain name that contains the full name of a host. For example, the domain name: challenger.ate.tbda.edu.

Partially Qualified Domain Name

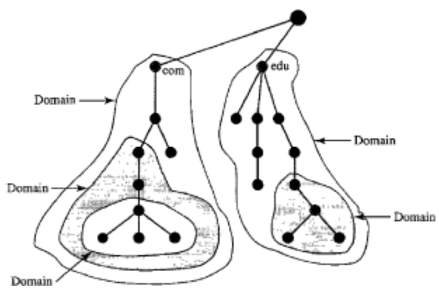
If a label is not terminated by a null string, it is called a partially qualified domain name (PQDN). A PQDN starts from a node, but it does not reach the root. It is used when the name to be resolved belongs to the same site as the client. For example, if a user at the *jhda.edu.* site wants to get the IP address of the challenger computer, he or she can define the partial name

Figure 25.4 shows some FQDNs and PQDNs.

Figure 25.4 *FQDN and PQDN*

FQDN	PQDN
challenger.atc.fhda.edu.	challenger.atc.fhda.edu
cs.hmune.com.	cs.hmune
www.funny.int.	www

Domain: A **domain** is a subtree of the domain name space. The name of the domain is the domain name of the node at the top of the subtree. Figure 25.5 shows some domains. Note that a domain may itselfbe divided into domains (or **subdomains** as they are sometimes called).

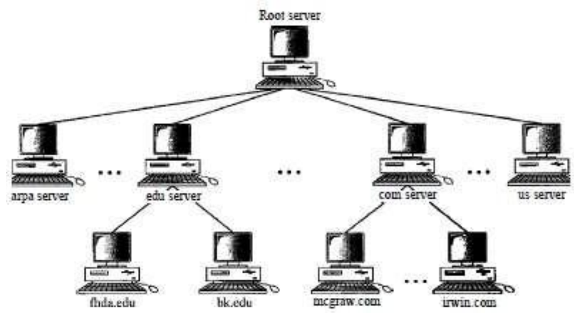


DISTRIBUTION OF NAME SPACE

The information contained in the domain name space must be stored. However, it is very inefficient and also unreliable to have just one computer store such a huge amount of information. It is inefficient because responding to requests from all over the world places a heavy load on the system. It is not unreliable because any failure makes the data inaccessible.

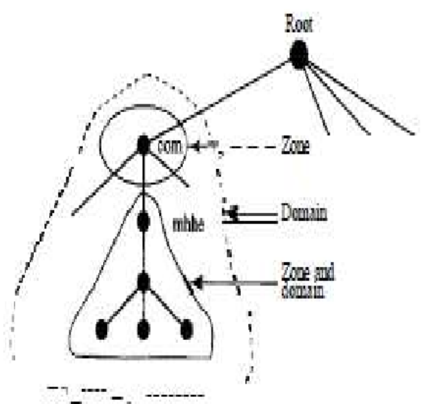
Hierarchy of Name Servers

The solution to these problems is to distribute the information among many computers called DNS servers. One way to do this is to divide the whole space into many domains based on the first level. In other words, we let the root stand alone and create as many domains (subtrees) as there are first-level nodes. Because a domain created in this way could be very large, DNS allows domains to be divided further into smaller domains (subdomains). Each server can be responsible (authoritative) for either a large or a small domain. In other words, we have a hierarchy of servers in the same way that we have a hierarchy of names (see Figure 25.6).



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. We can define a zone as a contiguous part of the entire tree. If a server accepts responsibility for a domain and does not divide the domain into smaller domains, the *domain* and the *zone* refer to the same thing. The server makes a database called a *zone file* and keeps all the information for every node under that domain. However, if a server divides its domain into subdomains and delegates part of its authority to other servers, *domain* and *zone* refer to different things. The information about the nodes in the subdomains is stored in the servers at the lower levels, with the original server keeping some sort of reference to these lower-level servers. Of course the original server does not free itself from responsibility totally: It still has a zone, but the detailed information is kept by the lower-level servers (see Figure 25.7). A server can also divide part of its domain and delegate responsibility but still keep part of the domain for itself. In this case, its zone is made of detailed information for the part of the domain that is not delegated and references to those parts that are delegated.



Root Server

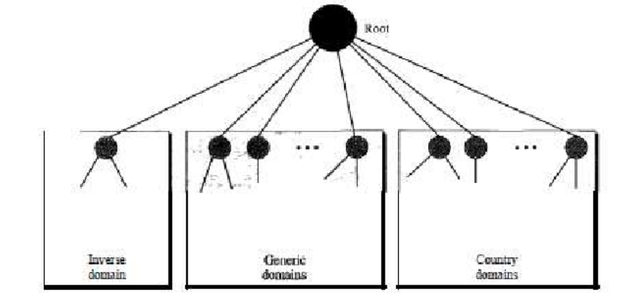
A root server is a server whose zone consists of the whole tree. A root server usually does not store any information about domains but delegates its authority to other servers, keeping references to those servers. There are several root servers, each covering the whole domain name space. The servers are distributed all around the world.

Primary and Secondary Servers

DNS defines two types of servers: primary and secondary. A primary server is a server that 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. A secondary server is a server that 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. If updaJing is required, it must be done by the primary server, which sends the updated version to the secondary. The primary and secondary servers are both authoritative for the zones they serve. The idea is not to put the secondary server at a lower level of authority but to create redundancy for the data so that if one server fails, the other can continue serving clients. Note also that a server can be a primary server for a specific zone and a secondary server for another zone. Therefore, when we refer to a server as a primary or secondary server, we should be careful to which zone we refer

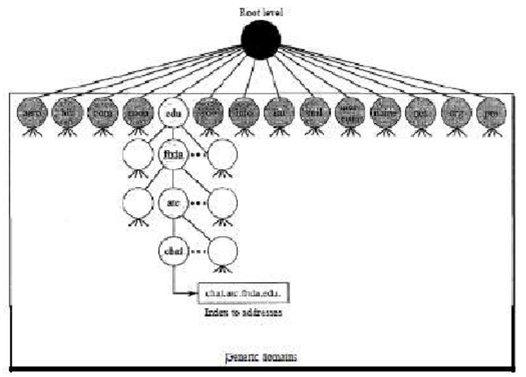
2. DNS IN THE INTERNET

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain.



Generic Domains

The **generic domains** define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database (see Figure 25.9).



Looking at the tree, we see that the first level in the generic domains section allows 14 possible labels. These labels describe the organization types as listed in Table 25.1.

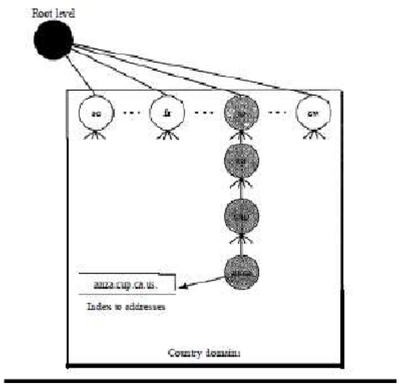
Table 25.1 *Generic domain labels*

Label	*Description*
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to "com")
com	Commercial organizations
coop	Cooperative business organizations
edu	Educational institutions
gov	Government institutions
info	Information service providers
int	International organizations
mil	Military groups
museum	Museums and other nonprofit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

Country Domains

The country domains section uses two-character country abbreviations (e.g., us for United States). Second labels can be organizational, or they can be more specific, national designations. The United States, for example, uses state abbreviations as a subdivision of us (e.g., ca.us.).

figure 25.10 shows the country domains section. The address *anza.cup.ca.us* can be translated to De Anza College in Cupertino, California, in the United States.



Inverse Domain

The inverse domain is used to map an address to a name. This may happen, for example, when a server has received a request from a client to do a task. Although the server has a file that contains a list of authorized clients, only the IP address of the client (extracted from the received IP packet) is listed. The server asks its resolver to send a query to the DNS server to map an address to a name to determine if the client is on the authorized list. This type of query is called an inverse or pointer (PTR) query. To handle a pointer query, the inverse domain is added to the domain name space with the first-level node called *arpa* (for historical reasons). The second level is also one single node named *in-addr* (for inverse address). The rest of the domain defines IP addresses. The servers that handle the inverse domain are also hierarchical. This means the netid part of the address should be at a higher level than the subnetid part, and the subnetid part higher than the hostid part. In this way, a server serving the whole site is at a higher level than the servers serving each subnet. This configuration makes the domain look inverted when compared to a generic or country domain. To follow the convention of reading the domain labels from the bottom to the top, an IF address such as 132.34.45.121 (a class B address with netid 132.34) is read as 121.45.34.132.in-addr. arpa. See Figure 25.11 for an illustration of the inverse domain configuration.

3. <u>ELECTRONIC MAIL</u>

One of the most popular Internet services is electronic mail (e-mail). The designers of the Internet probably never imagined the popularity of this application program. At the beginning of the Internet era, the messages sent by electronic mail were short and consisted of text only; they let people exchange quick memos. Today, electronic mail is much more complex. It allows a message to include text, audio, and video. It also allows one message to be sent to one or more recipients.

Architecture

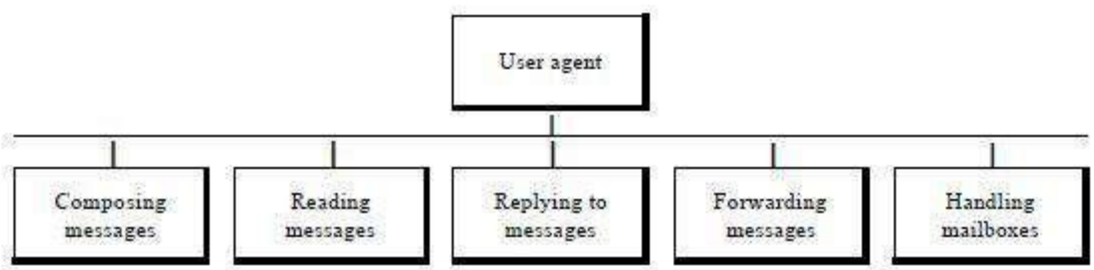
User Agent

The first component of an electronic mail system is the user agent. It provides service to the user to make the process of sending and receiving a message easier.

Services Provided by a User Agent

A user agent is a software package (program) that composes, reads, replies to, and forwards messages. It also handles mailboxes. Figure 26.11 shows the services of a typical user agent.

Figure 26.11 *Services ofuser agent*



Composing Messages

A user agent helps the user compose the e-mail message to be sent out. Most user agents provide

a template on the screen to be filled in by the user. Some even have a built-in editor that can do

spell checking, grammar checking, and other tasks expected from a sophisticated word processor. A user, of course, could alternatively use his or her favorite text editor or word processor to create the message and import it, or cut and paste it, into the user agent template.

Reading Messages

The second duty of the user agent is to read the incoming messages. When a user invokes a user agent, it first checks the mail in the incoming mailbox. Most user agents show a one-line summary of each received mail. Each e-mail contains the following fields.

1. A number field.

2. A flag field that shows the status of the mail such as new, already read but not replied to, or read and replied to.

3. The size of the message.

4. The sender.

5. The optional subject field.

Replying to Messages

After reading a message, a user can use the user agent to reply to a message. A user agent usually allows the user to reply to the original sender or to reply to all recipients of the message. The reply message may contain the original message and the new message.

Forwarding Messages

Replying is defined as sending a message to the sender or recipients of the copy. *Forwarding* is defined as sending the message to a third party. A user agent allows the receiver to forward the message, with or without extra comments, to a third party.

Handling Mailboxes

A user agent normally creates two mailboxes: an inbox and an outbox. Each box is a file with a special format that can be handled by the user agent. The inbox keeps all the received e-mails until they are deleted by the user. The outbox keeps all the sent e-mails until the user deletes them. Most user agents today are capable of creating customized mailboxes.

User Agent Types

There are two types of user agents: command-driven and GUI-based.

Command-Driven: Command-driven user agents belong to the early days of electronic mail. They are still present as the underlying user agents in servers. A command-driven user agent normally accepts a one-character command from the keyboard to perform its task. For example,

a user can type the character r, at the command prompt, to reply to the sender of the message, or type the character R to reply to the sender and all recipients. Some examples of command-driven user agents are *mail, pine,* and *elm.*

GUI-Based: Modern user agents are GUI-based. They contain graphical-user interface (GUI) components that allow the user to interact with the software by using both the keyboard and the mouse. They have graphical components such as icons, menu bars, and windows that make the services easy to access. Some examples of GUI-based user agents are Eudora, Microsoft's Outlook, and Netscape.

Addresses

To deliver mail, a mail handling system must use an addressing system with unique addresses. In the Internet, the address consists of two parts: a local part and a domain name, separated by an @

sign

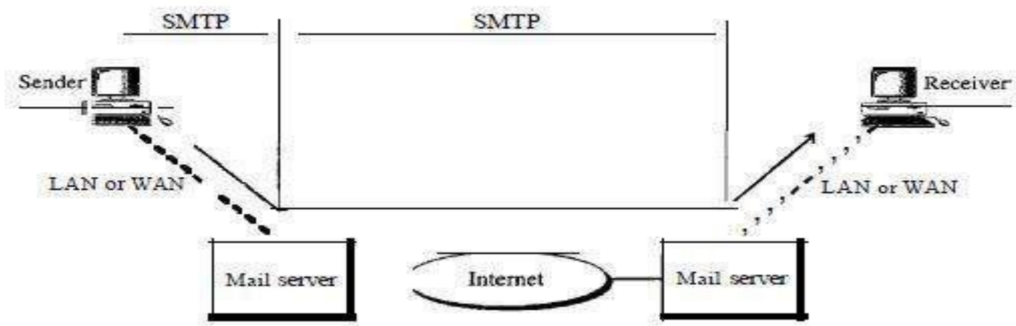
Local Part: The local part defines the name of a special file, called the user mailbox, where all the mail received for a user is stored for retrieval by the message access agent.

Domain Name: The second part of the address is the domain name. An organization usually selects one or more hosts to receive and send e-mail; the hosts are sometimes called *mail servers* or *exchangers.* The domain name assigned to each mail exchanger either comes from the DNS database or is a logical name (for example, the name of the organization).

SMTP

The actual mail transfer is done through message transfer agents. To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA. The formal protocol that defines the MTA client and server in the Internet is called the Simple Mail Transfer Protocol (SMTP).

Figure 26.16 *SMTP range*

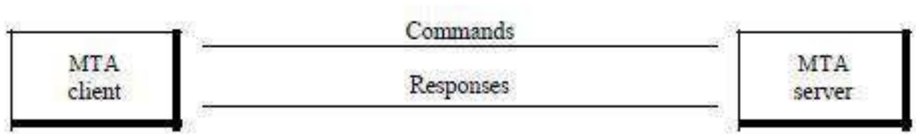


SMTP is used two times, between the sender and the sender's mail server and between the two mail servers. SMTP simply defines how commands and responses must be sent back and forth.

Commands and Responses

SMTP uses commands and responses to transfer messages between an MTA client and an MTA server (see Figure 26.17).

Figure 26.17 *Commands and responses*



Commands: Commands are sent from the client to the server. The format of a command is shown in Figure 26.18. It consists of a keyword followed by zero or more arguments. SMTP defines 14 commands. The first five are mandatory; every implementation must support these five commands. The next three are often used and highly recommended. The last six are seldom used.

Figure 26.18 *Commandformat*

Keyword: argument(s)

The commands are listed in Table 26.7.

Table 26.7 *Commands*

Keyword	*Argument(s)*
HELO	Sender's host name
MAIL FROM	Sender of the message
RCPTTO	Intended recipient of the message
DATA	Body of the mail
QUIT	
RSET	
VRFY	Name of recipient to be verified
NOOP	
TURN	
EXPN	Mailing list to be expanded
HELP	Command name

Responses: Responses are sent from the server to the client. A response is a three digit code that may be followed by additional textual information. Table 26.8 lists some of the responses.

Table 26.8 *Responses*

Code	*Description*
	Positive Completion Reply
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
	Positive Intermediate Reply
354	Start mail input

Mail Transfer Phases

The process of transferring a mail message occurs in three phases: connection establishment, mail transfer, and connection termination.

POP3

Post Office Protocol, version 3 (POP3) is simple and 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 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. The delete mode is normally used when the user is working at her permanent computer and can save and organize the received mail after reading or replying. The keep mode is normally used when the user accesses her mail away from her primary computer (e.g., a laptop). The mail is read but kept in the system for later retrieval and organizing.

IMAP

Another mail access protocol is Internet Mail Access Protocol, version 4 (IMAP4). IMAP4 is similar to POP3, but it has more features; IMAP4 is more powerful and more complex.

POP3 is deficient in several ways. It does not allow the user to organize her mail on the server; the user cannot have different folders on the server. In addition, POP3 does not allow the user to partially check the contents of the mail before downloading. IMAP4 provides the following extra functions:

o A user can check the e-mail header prior to downloading.

o A user can search the contents of the e-mail for a specific string of characters prior to downloading.

o A user can partially download e-mail. This is especially useful if bandwidth is limited and the e-mail contains multimedia with high bandwidth requirements.

o A user can create, delete, or rename mailboxes on the mail server.

o A user can create a hierarchy of mailboxes in a folder for e-mail storage.

2. FTP

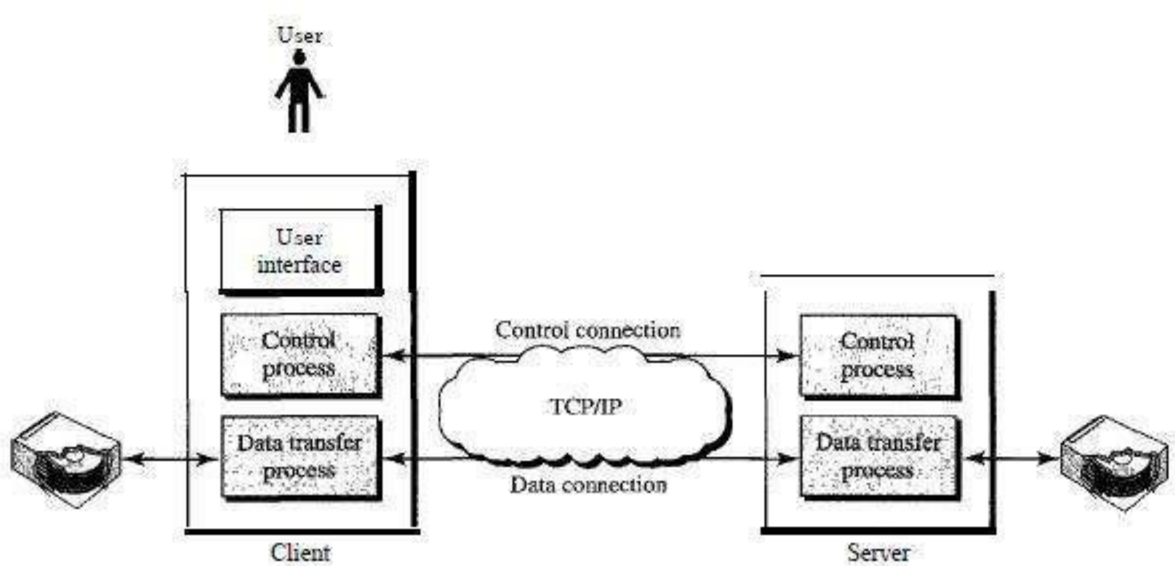
File Transfer Protocol (FTP) is the standard mechanism provided by *TCP/IP* for copying a file from one host to another. Although transferring files from one system to another seems simple and straightforward, some problems must be dealt with first. For example, two systems may use different file name conventions. Two systems may have different ways to represent text and data.

Two systems may have different directory structures. All these problems have been solved by FTP in a very simple and elegant approach.

FTP differs from other client/server applications in that it establishes two connections between the hosts. One connection is used for data transfer, the other for control information (commands and responses). Separation of commands and data transfer makes FTP more efficient. The control connection uses very simple rules of communication. We need to transfer only a line of command or a line of response at a time. The data connection, on the other hand, needs more complex rules due to the variety of data types transferred. However, the difference in complexity is at the FTP level, not TCP. For TCP, both connections are treated the same. FTP uses two well-known TCP ports: Port 21 is used for the control connection, and port 20 is used for the data connection.

Figure 26.21 shows the basic model of FTP. The client has three components: user interface, 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.

Figure 26.21 *FTP*



The control connection remains connected during the entire interactive FTP session. The data connection is opened and then closed for each file transferred. It opens each time commands that involve transferring files are used, and it closes when the file is transferred. In other words, 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.

Transmission Mode: FTP can transfer a file across the data connection by using one of the following three transmission modes: stream mode, block mode, and compressed mode. The stream mode is the default mode. Data are delivered from FTP to TCP as a continuous stream of bytes. TCP is responsible for chopping data into segments of appropriate size. If the data are simply a stream of bytes (file structure), no end-of-file is needed. End-of-file in this case is the closing of the data connection by the sender. If the data are divided into records (record structure), each record will have a 1-byte end-of-record (EOR) character and the end of the file will have a 1-byte end-of-file (EOF) character. In block mode, data can be delivered from FTP to TCP in blocks. In this case, each block is preceded by a 3-byte header. The first byte is called the *block descriptor;* the next 2 bytes define the size of the block in bytes. In the compressed mode, if the file is big, the data can be compressed. The compression method normally used is run-length encoding. In this method, consecutive appearances of a data unit are replaced by one occurrence and the number of repetitions. In a text file, this is usually spaces (blanks). In a binary file, null characters are usually compressed.

3. HTTP

The Hypertext Transfer Protocol (HTTP) is a protocol used mainly to access data on the World Wide Web. HTTP functions as a combination of FTP and SMTP. It is similar to FTP because it transfers files and uses the services of TCP. However, it is much simpler than FTP because it uses only one TCP connection. There is no separate control connection; only data are transferred between the client and the server. HTTP is like SMTP because the data transferred between the client and the server look like SMTP messages. In addition, the format of the messages is controlled by MIME-like headers. Unlike SMTP, the HTTP messages are not destined to be read by humans; they are read and interpreted by the HTTP server and HTTP client (browser). SMTP messages are stored and forwarded, but HTTP messages are delivered immediately. The commands from the client to the server are embedded in a request message. The contents of the requested file or other information are embedded in a response message. HTTP uses the services of TCP on well-known port 80.

HTTP Transaction

Figure 27.12 illustrates the HTTP transaction between the client and server. Although HTTP uses the services of TCP, HTTP itself is a stateless protocol. The client initializes the transaction by sending a request message. The server replies by sending a response.

Messages

The formats of the request and response messages are similar; both are shown in Figure 27.13. A request message consists of a request line, a header, and sometimes a body. A response message consists of a status line, a header, and sometimes a body.

Figure 27.12 *HTTP transaction*

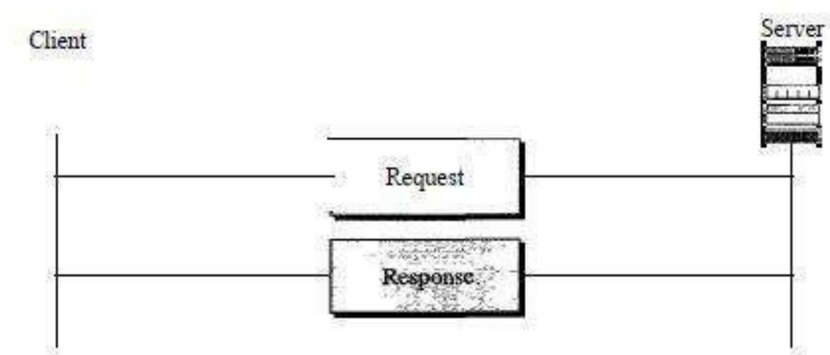
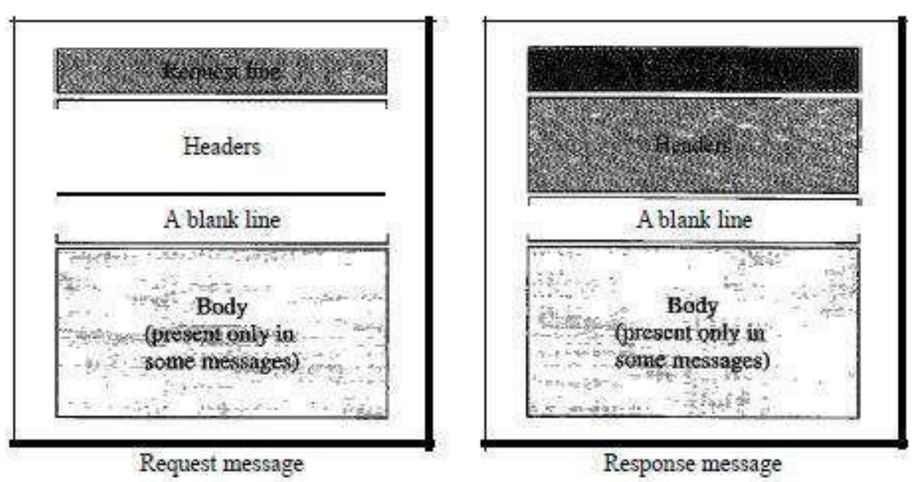
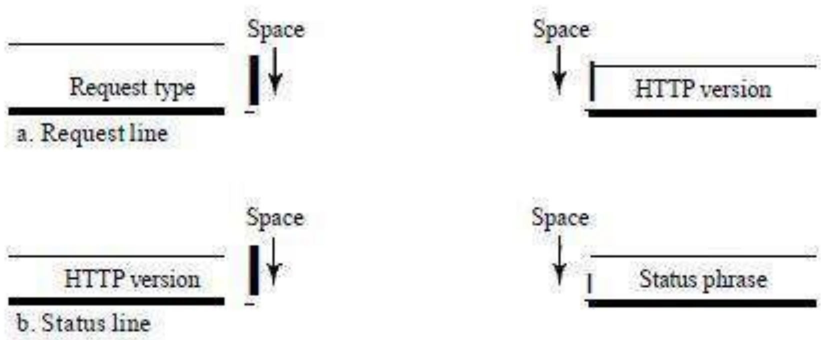


Figure 27.13 *Request and response messages*



Request and Status Lines: The first line in a request message is called a request line; the first line in the response message is called the status line. There is one common field, as shown in Figure 27.14.

Figure 27.14 *Request and status lines*



a. **Request type**. This field is used in the request message. In version 1.1 of HTTP, several request types are defined. The request type is categorized into *methods* as defined in Table 27.1.

Table 27.1 *Methods*

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

2. **URL**. URL means Uniform Resource Locator

3. **Version**. The current version of HTTP

4. **Status code**. This field is used in the response message. The status code field is similar to those in the FTP and the SMTP protocols. It consists of three digits. Whereas the codes in the 100 range are only informational, the codes in the 200 range indicate a successful request. The codes in the 300 range redirect the client to another URL, and the codes in the 400 range indicate

an error at the client site. Finally, the codes in the 500 range indicate an error at the server site. We list the most common codes in Table 27.2.

e. **Status phrase**. This field is used in the response message. It explains the status code in text form. Table 27.2 also gives the status phrase.

Table 27.2 *Status codes*

Code	*Phrase*	*Description*
		Informational
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
		Success
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

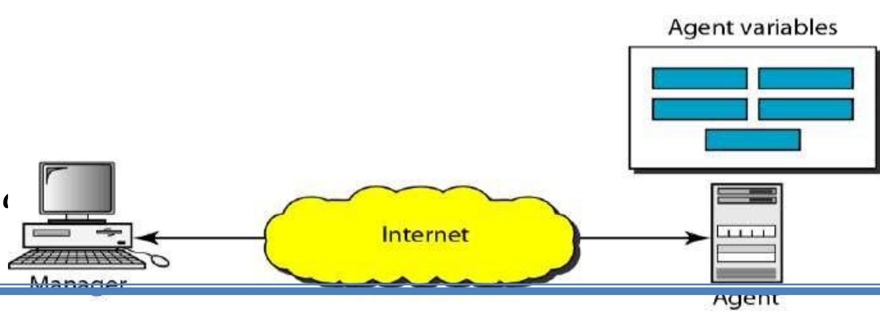
Simple Network Management Protocol (SNMP)

The Simple Network Management Protocol (SNMP) is a framework for managing devices in an internet using the TCPIIP protocol suite. It provides a set of fundamental operations for monitoring and maintaining an internet.

Concept

SNMP uses the concept of manager and agent. That is, a manager, usually a host, controls and monitors a set of agents, usually routers (see Figure below). SNMP is an application-level protocol in which a few manager stations control a set of agents. The protocol is designed at the application level so that it can monitor devices made by different manufacturers and installed on different physical networks.

Managers



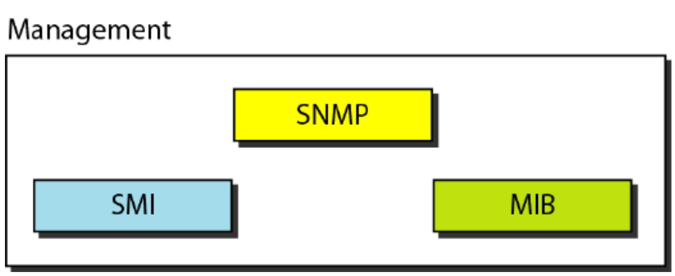
A management station, called a manager, is a host that runs the SNMP client program. A managed station, called an agent, is a router (or a host) that runs the SNMP server program. Management is achieved through simple interaction between a manager and an agent. The agent keeps performance information in a database. The manager has access to the values in the database. Agents can also contribute to the management process. The server program running on the agent can check the environment, and if it notices something unusual, it can send a warning message, called a trap, to the manager. warning message, called a trap, to the manager.

In other words, management with SNMP is based on three basic ideas:

1. A manager checks an agent by requesting information that reflects the behavior of the agent.
2. A manager forces an agent to perform a task by resetting values in the agent database.
3. An agent contributes to the management process by warning the manager of an unusual situation.

Management Components

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



RoleofSNMP

SNMP has some very specific roles in network management. SNMP defines the format of packets exchanged between a manager and an agent. It reads and changes the status (values) of objects (variables) in SNMP packets.

Role ofSMI

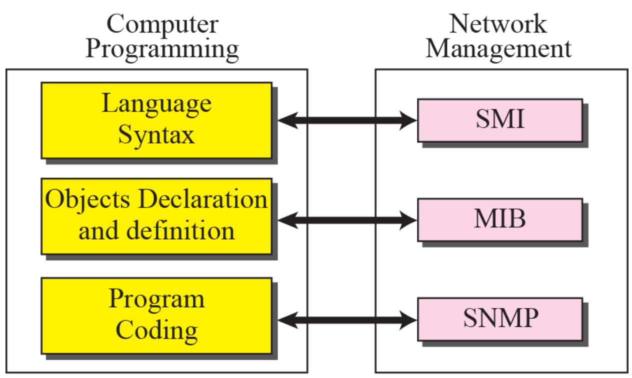
SMI defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values. SMI does not define the number of objects an entity should manage or name the objects to be managed or define the association between the objects and their values.

Role ofMIB

We need some protocol to define the number of objects, name them according to the rules defined by SMI, and associate a type to each named object and thus the role of MIB creates a collection of named objects, their types, and their relationships to each other in an entity to be managed.

ANALOGY

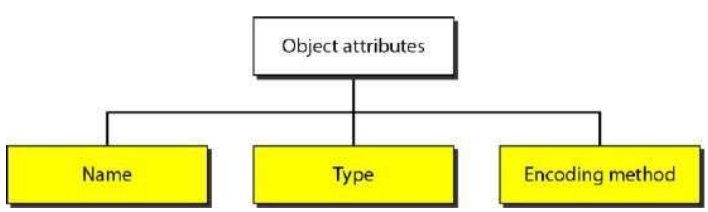
The functions of SNMP is similar to the functions of any of the programming. The role of SMI is similar to the rules followed by the programming language. The job of MIB lies in defining and declaring the various objects.



Structure of Management Information

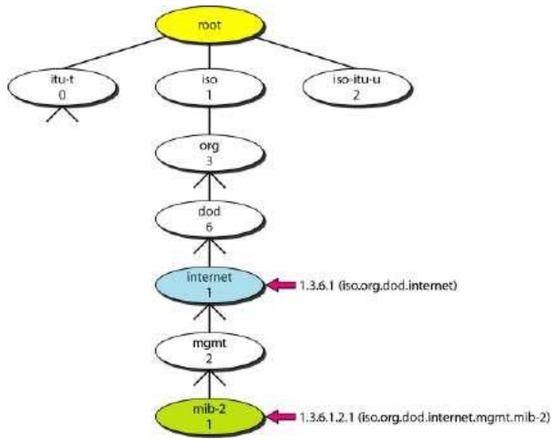
The Structure of Management Information, version 2 (SMIv2) is a component for network management. Its functions are

1. To name objects
2. To define the type of data that can be stored in an object
3. To show how to encode data for transmission over the network



1. Naming of Objects:

SMI requires that each managed object (such as a router, a variable in a router, a value) have a unique name. To name objects globally, SMI uses an object identifier, which is a hierarchical identifier based on a tree structure (see Figure below).

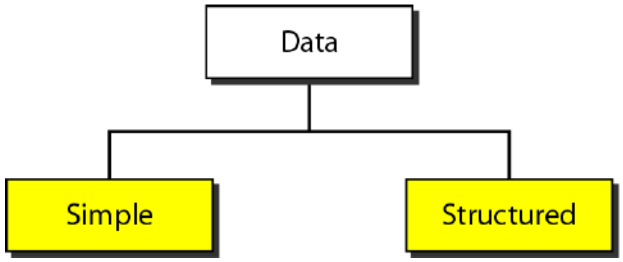


The tree structure starts with an unnamed root. Each object can be defined by using a sequence of integers separated by dots. The tree structure can also define an object by using a sequence of textual names separated by dots. The integer-dot representation is used in SNMP. The name-dot notation is used by people. For example, the following shows the same object in two different notations:

iso.org.dod.internet.mgmt.mib-2 ... 1.3.6.1.2.1

2. Type of the data:

The second attribute of an object is the type of data stored in it. SMI has two broad categories of data type: *simple* and *structured*



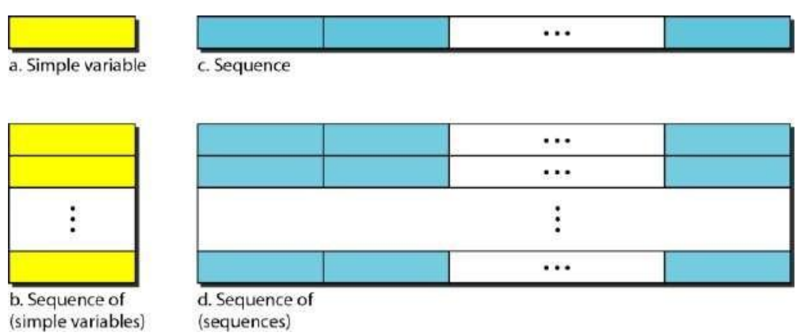
Simple Type The simple data types are atomic data types. Some of them are taken directly from ASN.l; others are added by SMI. Table 28.1. The first five are from ASN.l; the next seven are defined by SMI.

Type	*Size*	*Description*
INTEGER	4 bytes	An integer with a value between -2^{31} and $2^{31} - 1$
Integer32	4 bytes	Same as INTEGER
Unsigned32	4 bytes	Unsigned with a value between 0 and $2^{32} - 1$
OCTET STRING	Variable	Byte string up to 65,535 bytes long
OBJECT IDENTIFIER	Variable	An object identifier
IPAddress	4 bytes	An IP address made of four integers
Counter32	4 bytes	An integer whose value can be incremented from 0 to 2^{32}; when it reaches its maximum value, it wraps back to 0.
Counter64	8 bytes	64-bit counter
Gauge32	4 bytes	Same as Counter32, but when it reaches its maximum value, it does not wrap; it remains there until it is reset
TimeTicks	4 bytes	A counting value that records time in $\frac{1}{100}$ s
BITS		A string of bits
Opaque	Variable	Uninterpreted string

Structured Type By combining simple and structured data types, we can make new structured data types. SMI defines two structured data types: *sequence* and *sequence of*

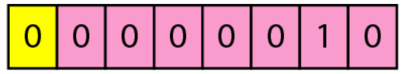
Sequence. A *sequence* data type is a combination of simple data types, not necessarily of the same type. It is analogous to the concept of a *struct* or a *record* used in programming languages such as C.

Sequence of. A *sequence of*data type is a combination of simple data types all of the same type or a combination of sequence data types all of the same type. It is analogous to the concept of an *array* used in programming languages such as C.

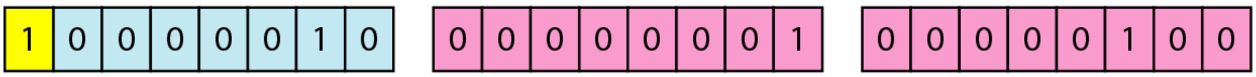


3. Encoding Method

SMI uses another standard, 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, as illustrated in Figure below.



a. The colored part defines the length (2).



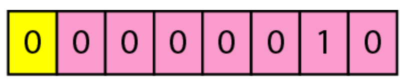
b. The shaded part defines the length of the length (2 bytes);
the colored bytes define the length (260 bytes).

Tag. The tag is a I-byte field that defines the type of data. It is composed of three subfields: *class* (2 *bits),fonnat* (1 bit), and *number* (5 bits). The class subfield defines the scope of the data. Four classes are defined: universal (00), applicationwide (01), context-specific (10), and private (11). The universal data types are those taken from ASN.l (INTEGER, OCTET STRING, and ObjectIdentifier). The applicationwide data types are those added by SMI (IPAddress, Counter, Gauge, and TimeTicks). The five context-specific data types have meanings that may change from one protocol to another. The private data types are vendor-specific.

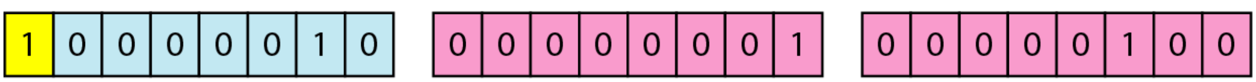
The **format** subfield indicates whether the data are simple (0) or structured (1). The number subfield further divides simple or structured data into subgroups. For example, in the universal class, with simple format, INTEGER has a value of 2, OCTET STRING has a value of 4, and so on. Table 28.2 shows the data types we use in this chapter and their tags in binary and hexadecimal numbers.

Data Type	Class	Format	Number	Tag (Binary)	Tag (Hex)
INTEGER	00	0	00010	**00000010**	**02**
OCTET STRING	00	0	00100	**00000100**	**04**
OBJECT IDENTIFIER	00	0	00110	**00000110**	**06**
NULL	00	0	00101	**00000101**	**05**
Sequence, sequence of	00	1	10000	**00110000**	**30**
IPAddress	01	0	00000	**01000000**	**40**
Counter	01	0	00001	**01000001**	**41**
Gauge	01	0	00010	**01000010**	**42**
TimeTicks	01	0	00011	**01000011**	**43**
Opaque	01	0	00100	**01000100**	**44**

Length. The length field is I or more bytes. If it is 1 byte, the most significant bit must be O. The other 7 bits define the length of the data. If it is more than 1 byte, the most significant bit of the first byte must be 1. The other 7 bits of the first byte define the number of bytes needed to define the length. See Figure 28.10 for adepiction of the length field.



a. The colored part defines the length (2).



b. The shaded part defines the length of the length (2 bytes);
the colored bytes define the length (260 bytes).

Management Information Base (MIB)

The Management Information Base, version 2 (MIB2) is the second component used in network management. Each agent has its own MIB2, which is a collection of all the objects that the manager can manage. The objects in MIB2 are categorized under 10 different groups as shown in figure.

sys This object *(system)* defines general information about the node (system), such as the name, location, and lifetime.

if This object *(inteiface)* defines information about all the interfaces of the node including interface number, physical address, and IP address.

at This object *(address translation)* defines the information about the ARP table.

ip This object defines information related to IP, such as the routing table and the IP address.

icmp This object defines information related to ICMP, such as the number of packets sent and received and total errors created.

tcp This object defines general information related to TCP, such as the connection table, time-out value, number of ports, and number of packets sent and received.

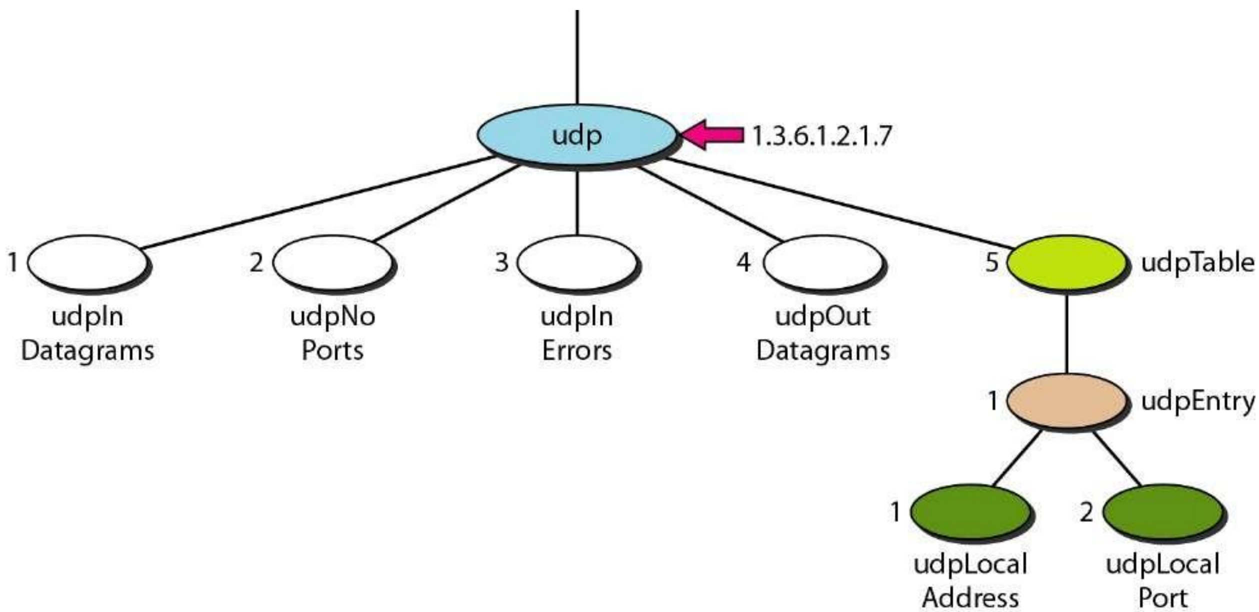
udp This object defines general information related to UDP, such as the number of ports and number of packets sent and received.

snmp This object defines general information related to SNMP itself.

Accessing MIB Variables

To show how to access different variables, we use the udp group as an example. There are four simple variables in the udp group and one sequence of (table of) records. Figure 28.16 shows the variables and the table. We will show how to access each entity.

Simple Variables To access any of the simple variables, we use the id of the group (1.3.6.1.2.1.7) followed by the id of the variable. The following shows how to access each variable.



udpInDatagrams.0	1.3.6.1.2.1.7.1.0
udpNoPorts.0	1.3.6.1.2.1.7.2.0
udpInErrors.0	1.3.6.1.2.1.7.3.0
udpOutDatagrams.0	1.3.6.1.2.1.7.4.0

udpTable ... 1.3.6.1.2.1.7.5

udpEntry ... 1.3.6.1.2.1.7.5.1

udpLocalAddress 1.3.6.1.2.1.7.5.1.1
udpLocalPoo' 1.3.6.1.2.1.7.5.1.2

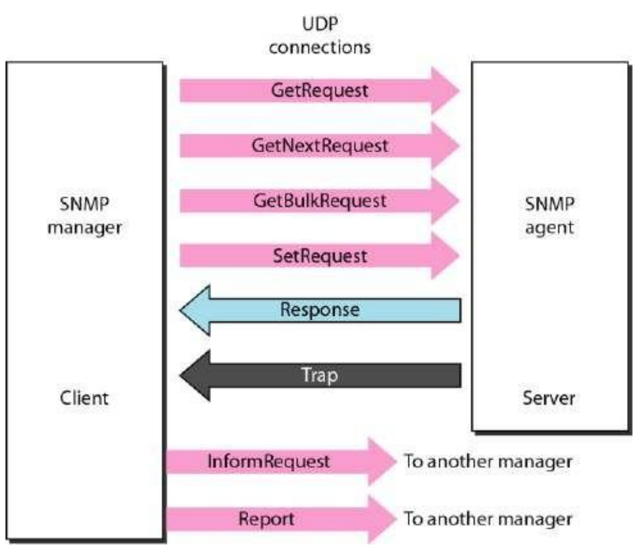
SNMP

SNMP uses both SMI and MIB in Internet network management. It is an application program that allows

1. A manager to retrieve the value of an object defined in an agent
2. A manager to store a value in an object defined in an agent
3. An agent to send an alarm message about an abnormal situation to the manager

PDUs

SNMPv3 defines eight types of packets (or PDUs): GetRequest, GetNextRequest, GetBulkRequest,
SetRequest, Response, Trap, InformRequest, and Report (see Figure below).



GetRequest The GetRequest PDU is sent from the manager (client) to the agent (server) to retrieve the value of a variable or a set of variables.

GetNextRequest The GetNextRequest PDU is sent from the manager to the agent to retrieve the value of a variable. The retrieved value is the value of the object following the defined Objectid in the PDD. It is mostly used to retrieve the values ofthe entries in a table. If the manager does not know the indexes of the entries, it cannot retrieve the values. However, it can use GetNextRequest and define the ObjectId of the table. Because the first entry has the ObjectId immediately after the ObjectId of the table, the value of the first entry is returned. The manager can use this ObjectId to get the value of the next one, and so on.

GetBulkRequest The GetBulkRequest POD is sent from the manager to the agent to retrieve a large amount of data. It can be used instead of multiple GetRequest and GetNextRequest PODs.

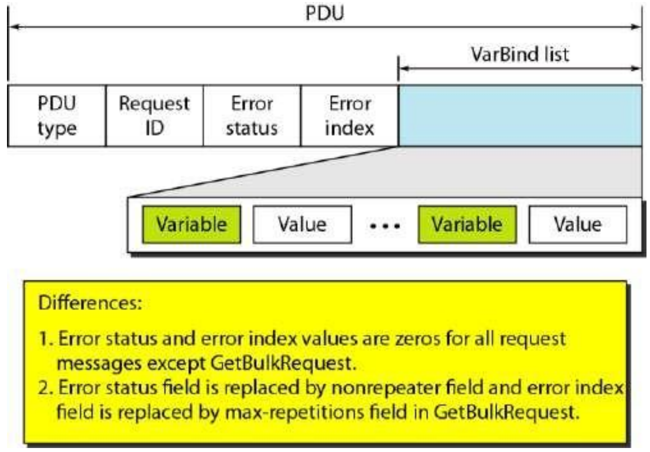
SetRequest The SetRequest PDD is sent from the manager to the agent to set (store) a value in a variable.

Response The Response PDD is sent from an agent to a manager in response to GetRequest or GetNextRequest. It contains the value(s) of the variable(s) requested by the manager.

Trap The Trap (also called SNMPv2 Trap to distinguish it from SNMPv1 Trap)

POD is sent from the agent to the manager to report an event. For example, if the agent is rebooted, it infonns the manager and reports the time of rebooting. InformRequest The InfonnRequest POD is sent from one manager to another remote manager to get the value of some variables from agents under the control of the remote manager. The remote manager responds with a Response POD.
Report The Report POD is designed to report some types of errors between managers. It is not yet in use.

Format
The fonnat for the eight SNMP PODs is shown in Figure 28.21. The GetBulkRequest POD differs from the others in two areas, as shown in the figure.



PDU type. This field defines the type of the POD (see Table 28.4).

Request ID. This field is a sequence number used by the manager in a Request POD and repeated by the agent in a response. It is used to match a request to a response. Error status. This is 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. Table 28.3 lists the types of errors that can occur.

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

Nonrepeaters. This field IS used only in GetBulkRequest and replaces the error status field, which is empty in Request PDUs.

Error index. The error index is an offset that tells the manager which variable caused the errOr.

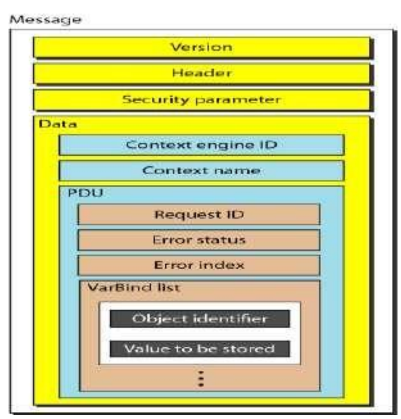
Max~repetition. This field is also used only in GetBulkRequest and replaces the error index field, which is empty in Request PDUs.

VarBind list. This is a set of variables with the corresponding values the manager wants to retrieve or set. The values are null in GetRequest and GetNextRequest. In a Trap PDU, it shows the variables and values related to a specific PDU.

Messages

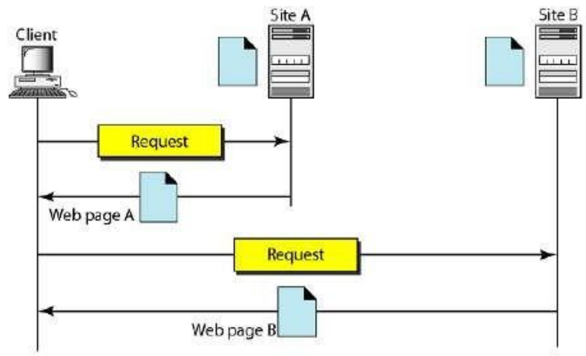
SNMP does not send only a PDU, it embeds the PDU in a message. A message in SNMPv3 is made of four elements: version, header, security parameters, and data (which include the encoded PDU), as shown in Figure 28.22.

Because the length of these elements is different from message to message, SNMP uses BER to encode each element. Remember that BER uses the tag and the length to define a value. The *version* defines the current version (3). The *header* contains values for message identification, maximum message size (the maximum size of the reply), message flag (one octet of data type OCTET STRING where each bit defines security type, such as privacy or authentication, Or other information), and a message security model (defining the security protocol). The message *security parameter* is used to create a message digest (see Chapter 31). The data contain the PDU. If the data are encrypted, there is information about the encrypting engine (the manager program that did the encryption) and the encrypting context (the type of encryption) followed by the encrypted PDU. If the data are not encrypted, the data consist of just the PDU. To define the type ofPDU, SNMP uses a tag. The class is context-sensitive (10), the format is structured (1), and the numbers are 0, 1,2, 3, 5, 6, 7, and 8 (see Table 28.4). Note that SNMPvl defined A4 for Trap, which is obsolete today.



WWW

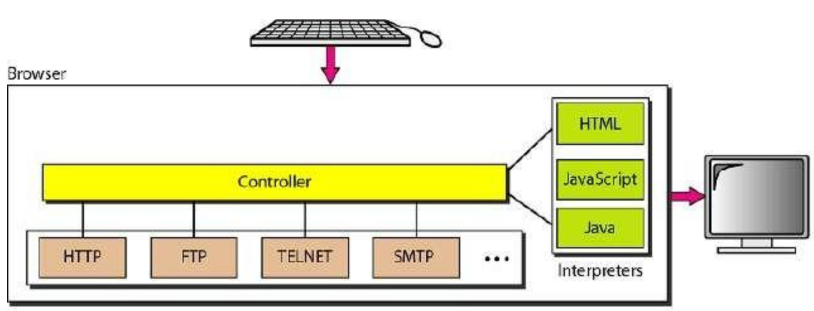
TheWWW today is a distributed clientJserver service, in which a client using a browser can access a service using a server. However, the service provided is distributed over many locations called *sites*, as shown **in** Figure below



Each site holds one or more documents, referred to as *Web pages.* Each Web page can contain a link to other pages in the same site or at other sites. The pages can be retrieved and viewed by using browsers. Let us go through the scenario shown in above Figure. The client needs to see some information that it knows belongs to site A. It sends a request through its browser, a program that is designed to fetch Web documents. The request, among other information, includes the address of the site and the Web page, called the URL, which we will discuss shortly. The server at site A finds the document and sends it to the client. When the user views the document, she finds some references to other documents, including a Web page at site B. The reference has the URL for the new site. The user is also interested in seeing this document. The client sends another request to the new site, and the new page is retrieved.

Client (Browser)

A variety of vendors offer commercial browsers that interpret and display a Web document,
and all use nearly the same architecture. Each browser usually consists of three parts: a controller, client protocol, 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.

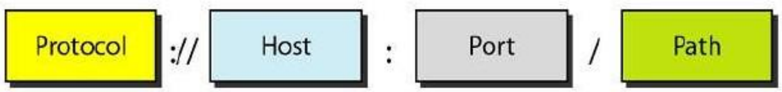


Server

The Web page is stored at the server. Each time a client 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 disk.

Uniform Resource Locator

A client that wants to access a Web page needs the address. To facilitate the access of documents distributed throughout the world, HTTP uses locators. The uniform resource locator (URL) is a standard for specifying any kind of information on the Internet. The URL defines four things: protocol, host computer, port, and path (see Figure below).



The ***protocol*** is the client/server program used to retrieve the document. Many different protocols can retrieve a document; among them are FTP or HTTP. The most common today is HTTP.

The **host** is the computer on which the information is located, although the name of the computer can be an alias. Web pages are usually stored in computers, and computers are given alias names that usually begin with the characters "www". This is not mandatory, however, as the host can be any name given to the computer that hosts the Web page. The URL can optionally contain the port number of the server.

If the ***port*** is included, it is inserted between the host and the path, and it is separated from the host by a colon.

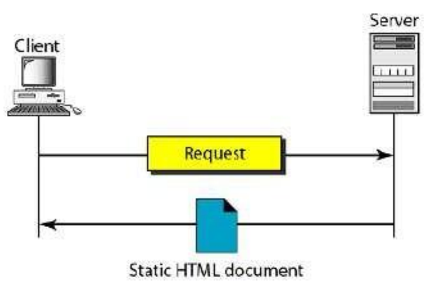
Path is the pathname of the file where the information is located. Note that the path can itself contain slashes that, in the UNIX operating system, separate the directories from the subdirectories and files.

WEB DOCUMENTS

The documents in the WWW can be grouped into three broad categories: static, dynamic, and active.

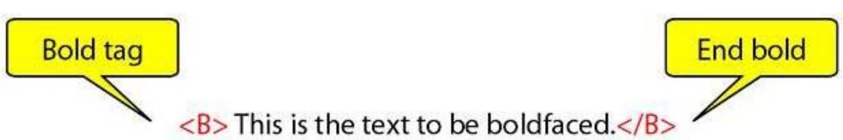
Static Documents

Static documents are fixed-content documents that are created and stored in a server. The client can get only a copy of the document. In other words, the contents of the file are determined when the file is created, not when it is used. Of course, the contents in the server can be changed, but the user cannot change them. When a client accesses the document, a copy of the document is sent. The user can then use a browsing program to display the document (see Figure below).

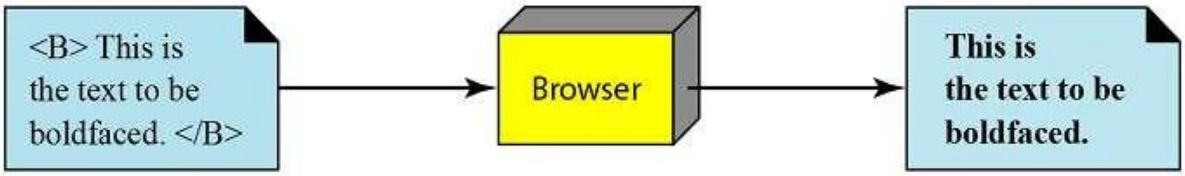


HTML

For creating web documents we use HTML. Hypertext Markup Language (HTML) is a language for creating Web pages. The term *markup language* comes from the book publishing industry. To make part of a text displayed in boldface with HTML, we put beginning and ending boldface tags (marks) in the text, as shown in Figure 27.5. The two tags <B> and </B> are instructions for the browser.

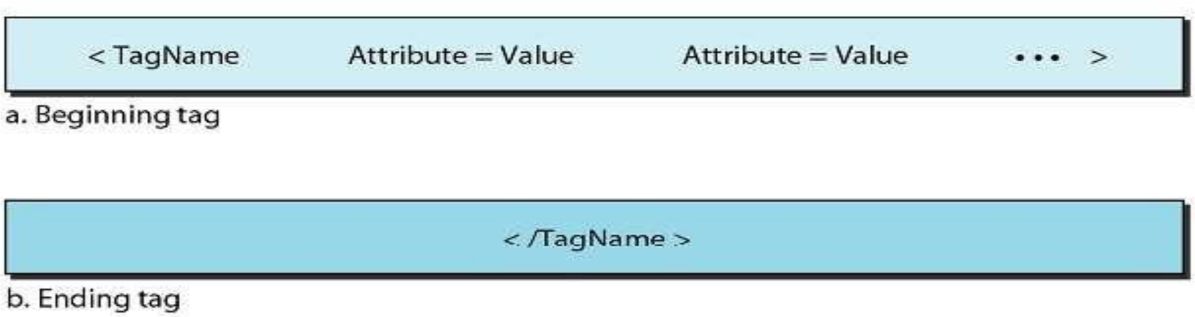


When the browser sees these two marks, it knows that the text must be boldfaced (see Figure above). A markup language such as HTML allows us to embed formatting instructions in the file itself. The instructions are included with the text. In this way, any browser can read the instructions and format the text according to the specific workstation



HTML lets us use only ASCII characters for both the main text and formatting instructions. In this way, every computer can receive the whole document as an ASCII document. The main text is the data, and the formatting instructions can be used by the browser to format the data. A Web page is made up of two parts: the head and the body. The head is the first part of a Web page. The head contains the title of the page and other parameters that the browser will use. The actual contents of a page are in the body, which includes the text and the tags. Whereas the text is the actual infonnation contained in a page, the tags define the appearance of the document. Every HTML tag is a name followed by an optional list of attributes, all enclosed between less-than and greater-than symbols « and >).

An **attribute**, if present, is followed by an equals sign and the value of the attribute. Some tags can be used alone; others must be used in pairs. Those that are used in pairs are called *beginning* and *ending* tags. The beginning tag can have attributes and values and starts with the name of the tag. The ending tag cannot have attributes or values but must have a slash before the name of the tag.

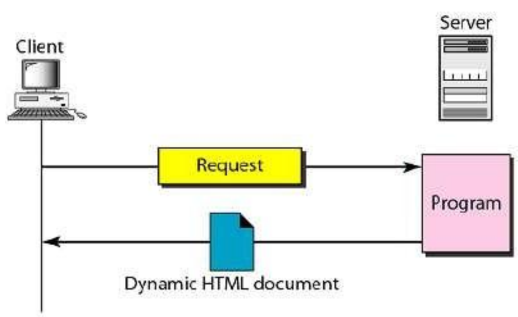


Dynamic Documents

A **dynamic document** 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 output of the program or script as a response to the browser that requested the document. Because a fresh document is created for each request, the contents of a dynamic document can vary from one request to another. very simple example of a dynamic document is the retrieval of the time and date from a server.

Common Gateway Interface (CGI)

The **Common** Gateway **Interface** (CGI) is a technology that creates and handles dynamic documents. CGI is a set of standards that defines how a dynamic document is written, how data are input to the program, and how the output result is used. CGI is not a new language; instead, it allows programmers to use any of several languages such as C, C++, Boume Shell, Kom Shell, C Shell, Tcl, or Perl. The only thing that CGI defines is a set of rules and tenns that the programmer must follow.



Input In traditional programming, when a program is executed, parameters can be passed to the program. Parameter passing allows the programmer to write a generic program that can be used in different situations

For ***example,*** *a generic copy program can be written to copy any file to another. A user can use the program to copy a file named x to another file named y by passing x and y as parameters. The input from a browser to a server is sent by using aform. If the infonnation in a fonn is small (such as a word), it can be appended to the URL after a question mark. For example, the following URL is carrying fonn infonnation (23, a value):*

http://www.deanzalcgi-binlprog.pl?23

When the server receives the URL, it uses the part of the URL before the question mark to access the program to be run, and it interprets the part after the question mark (23) as the input sent by *the client. It stores this string in a variable. When the CGI program is executed, it can access this value.*

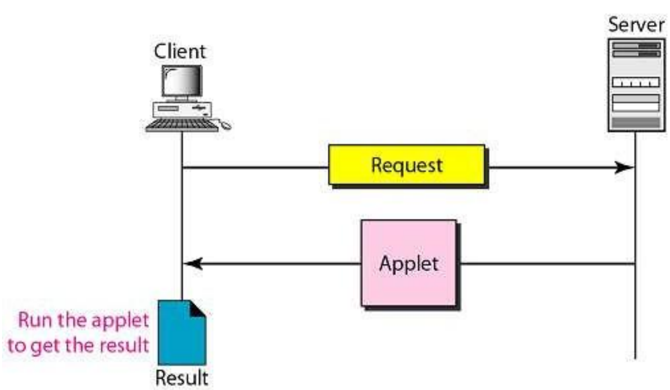
Output The whole idea of CGI is to execute a CGI program at the server site and send the output to the client (browser). The output is usually plain text or a text with HTML structures; however, the output can be a variety of other things. It can be graphics or binary data, a status code, instructions to the browser to cache the result, or instructions to the server to send an existing document instead of the actual output.

Active Documents

For many applications, we need a program or a script 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.

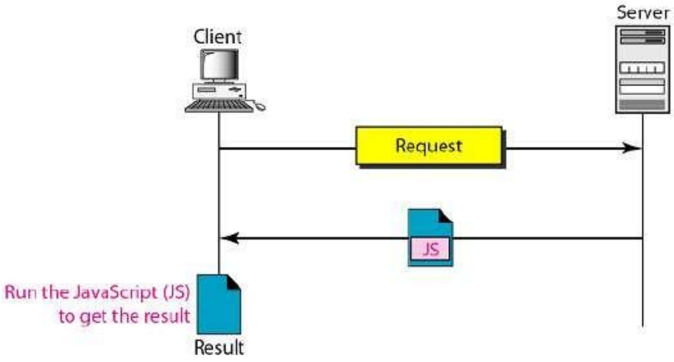
Java Applets

One way to create an active document is to use Java applets. Java is a combination of a high-level programming language, a run-time environment, and a class library that allows a programmer to write an active document (an applet) and a browser to run it. It can also be a stand-alone program that doesn't use a browser.



JavaScript

The idea of scripts in dynamic documents can also be used for active documents. If the active part of the document is small, it can be written in a scripting language; then it can be interpreted and run by the client at the same time and the same is illustrated in following fig.



HTTP:

HTTP stands for HyperText Transfer Protocol. Tim Berner invents it. HyperText is the type of text that is specially coded with the help of some standard coding language called HyperText Markup Language (HTML). **HTTP/2** is the new version of HTTP. HTTP/3 is the latest version of HTTP, which is published in 2022.

The protocol used to transfer hypertext between two computers is known as HyperText Transfer Protocol.

HTTP provides a standard between a web browser and a web server to establish communication. It is a set of rules for transferring data from one computer to another. Data such as text, images, and other multimedia files are shared on the World Wide Web. Whenever a web user opens their web browser, the user indirectly uses HTTP. It is an application protocol that is used for distributed, collaborative, hypermedia information systems.

Working of HTTP:

First of all, whenever we want to open any website then first open a web browser after that we will type the URL of that website (e.g., www.facebook.com). This URL is now sent to the Domain Name Server (DNS). Then DNS first check records for this URL in their database, then DNS will return the IP address to the web browser corresponding to this URL. Now the browser is able to send requests to the actual server.

After the server sends data to the client, the connection will be closed. If we want something else from the server we should have to re-establish the connection between the client and the server.

HTTP Connection

HTTP Request

HTTP request is simply termed as the information or data that is needed by Internet browsers for loading a website. This is simply known as HTTP Request.

There is some common information that is generally present in all HTTP requests. These are mentioned below.

- HTTP Version
- URL
- HTTP Method
- HTTP Request Headers
- HTTP Body

HTTP Request Headers

HTTP Request Headers generally store information in the form of key-value and must be present in each HTTP Request. The use of this Request Header is to provide core information about the client's information, etc.

HTTP Request Body

HTTP Request Body simply contains the information that has to be transferred. HTTP Request has the information or data to be sent to these browsers.

HTTP Method

HTTP Methods are simply HTTP Verb. In spite of being present so many HTTP Methods, the most common HTTP Methods are HTTP GET and HTTP POST. These two are generally used in HTTP cases. In HTTP GET, the information is received in the form of a website.

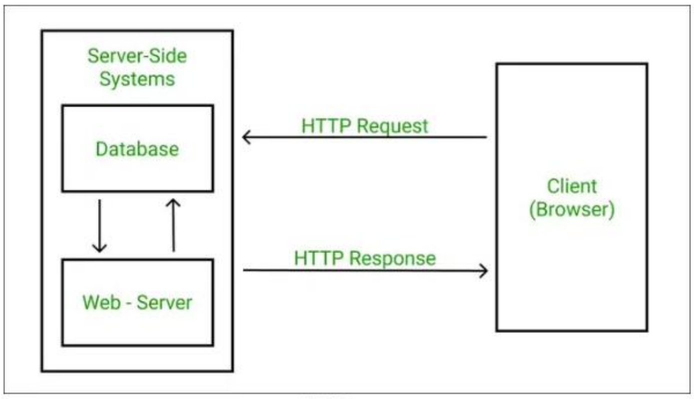
For more, refer to the Difference Between HTTP GET and HTTP POST.

HTTP Response

HTTP Response is simply the answer to what a Server gets when the request is raised. There are various things contained in HTTP Response, some of them are listed below.

- HTTP Status Code
- HTTP Headers

- HTTP Body



HTTP Response

HTTP Response Headers

HTTP Response headers are simply like an HTTP Request where it has that work to send some important files and data to the HTTP Response Body.

HTTP Response Body

HTTP Responses are the responses that are received successfully upon the request. Generally, it comes under the requests generated by the web. In most cases, the request is of transferring the HTML data into a webpage.

HTTP Status Code

HTTP Status Codes are the 3-Digit codes that tell the message or simply tell us about the HTTP Request whether it has been completed or not. There are simply 5 types of status codes.

- Informational
- Successful
- Re-directional
- Client-Error
- Server-Error

Characteristics of HTTP

HTTP is IP based communication protocol that is used to deliver data from server to client or vice-versa.

- The server processes a request, which is raised by the client, and also server and client know each other only during the current bid and response period.
- Any type of content can be exchanged as long as the server and client are compatible with it.
- Once data is exchanged, servers and clients are no longer connected.

- It is a request and response protocol based on client and server requirements.
- It is a connection-less protocol because after the connection is closed, the server does not remember anything about the client and the client does not remember anything about the server.
- It is a stateless protocol because both client and server do not expect anything from each other but they are still able to communicate.

Advantages of HTTP

- Memory usage and CPU usage are low because of fewer simultaneous connections.
- Since there are few TCP connections hence network congestion is less.
- Since handshaking is done at the initial connection stage, then latency is reduced because there is no further need for handshaking for subsequent requests.
- The error can be reported without closing the connection.
- HTTP allows HTTP pipe-lining of requests or responses.

Disadvantages of HTTP

- HTTP requires high power to establish communication and transfer data.
- HTTP is less secure because it does not use any encryption method like HTTPS and use TLS to encrypt regular HTTP requests and response.
- HTTP is not optimized for cellular phones and it is too gabby.
- HTTP does not offer a genuine exchange of data because it is less secure.
- The client does not close the connection until it receives complete data from the server; hence, the server needs to wait for data completion and cannot be available for other clients during this time.

File Transfer Protocol (FTP):

File Transfer Protocol(FTP) is an application layer protocol that moves files between local and remote file systems. It runs on top of TCP, like HTTP. To transfer a file, 2 TCP connections are used by FTP in parallel: control connection and data connection.

Types of FTP

There are different ways through which a server and a client do a file transfer using FTP. Some of them are mentioned below:

- **Anonymous FTP:** Anonymous FTP is enabled on some sites whose files are available for public access. A user can access these files without having any username or password. Instead, the username is set to anonymous, and the password is to the guest by default. Here, user access is very limited. For example, the user can be allowed to copy the files but not to navigate through directories.

- **Password Protected FTP:** This type of FTP is similar to the previous one, but the change in it is the use of username and password.
- **FTP Secure (FTPS):** It is also called as FTP Secure Sockets Layer (FTP SSL). It is a more secure version of FTP data transfer. Whenever FTP connection is established, Transport Layer Security (TLS) is enabled.
- **FTP over Explicit SSL/TLS (FTPES):** FTPES helps by upgrading FTP Connection from port 21 to an encrypted connection.
- **Secure FTP (SFTP):** SFTP is not a FTP Protocol, but it is a subset of Secure Shell Protocol, as it works on port 22.

Here are steps mentioned in which FTP works:

- A user has to log in to FTP Server first, there may be some servers where you can access to content without login, known as anonymous FTP.
- Client can start a conversation with server, upon requesting to download a file.
- The user can start different functions like upload, delete, rename, copy files, etc. on server.

Types of Connection in FTP

- Control Connection
- Data Connection

Control Connection

For sending control information like user identification, password, commands to change the remote directory, commands to retrieve and store files, etc., FTP makes use of a control connection. The control connection is initiated on port number 21.

Data connection

For sending the actual file, FTP makes use of a data connection. A data connection is initiated on port number 20.

FTP sends the control information out-of-band as it uses a separate control connection. Some protocols send their request and response header lines and the data in the same TCP connection. For this reason, they are said to send their control information in-band. HTTP and SMTP are such examples.

FTP Session

When an FTP session is started between a client and a server, the client initiates a control TCP connection with the server side. The client sends control information over this. When the server receives this, it initiates a data connection to the client side. But the control connection remains active throughout the user session. As we know HTTP is stateless . But FTP needs to maintain a state about its user throughout the session.

FTP Clients

FTP works on a client-server model. The FTP client is a program that runs on the user's computer to enable the user to talk to and get files from remote computers. It is a set of commands that establishes the connection between two hosts, helps to transfer the files, and then closes the connection.

Characteristics of FTP

- FTP uses TCP as a transport layer protocol.
- It is good for simple file transfers, such as during boot time.
- Errors in the transmission (lost packets, checksum errors) must be handled by the TFTP server.
- It uses only one connection through well-known port 69.
- TFTP uses a simple lock-step protocol (each data packet needs to be acknowledged). Thus the throughput is limited.

FTP's Security Issues

- Information could not go across a secure tunnel since FTP was not intended to do so. Thus, encryption is not present. A hacker would not need to struggle with encryption to access or alter data that is usable if they could intercept an FTP transaction.
- Even with FTP cloud storage, data can still be intercepted and misused if the service provider's system is attacked.
- As a result, data sent via FTP is a target for spoofing, sniffing, brute force, and other types of attacks that move somewhat slowly. A hacker might examine an FTP transmission and try to take advantage of any flaws by simply port scanning.
- The fact that FTP uses clear-text passwords—passwords that haven't been encrypted—is one of its main security flaws. Put differently, "Jerry1992" appears exactly like "Jerry1992." The real password is hidden via an algorithm in more secure protocols. As a result, "Jerry1992" might appear as "dj18387saksng8937d9d8d7s6a8d89." Passwords like this are not secured by FTP, which makes them more easily cracked by malicious actors.

Advantages of FTP

- File sharing also comes in the category of advantages of FTP in this between two machines files can be shared on the network.

- Speed is one of the main benefits of FTP.
- Since we don't have to finish every operation to obtain the entire file, it is more efficient.
- Using the username and password, we must log in to the FTP server. As a result, FTP might be considered more secure.
- We can move the files back and forth via FTP. Let's say you are the firm manager and you provide information to every employee, and they all reply on the same server.

Disadvantages of FTP

- File size limit is the drawback of FTP only 2 GB size files can be transferred.
- More then one receivers are not supported by FTP.
- FTP does not encrypt the data this is one of the biggest drawbacks of FTP.
- FTP is unsecured we use login IDs and passwords making it secure but they can be attacked by hackers.

TELNET:

TELNET stands for Teletype Network. It is a type of protocol that enables one computer to connect to the local computer. It is used as a standard TCP/IP protocol for virtual terminal service which is provided by ISO. The computer which starts the connection is known as the local computer.

The computer which is being connected to i.e. which accepts the connection known as the remote computer.

During telnet operation, whatever is being performed on the remote computer will be displayed by the local computer. Telnet operates on a client/server principle. The local computer uses a telnet client program and the remote computers use a telnet server program.

Logging

The logging process can be further categorized into two parts:

1. Local Login
2. Remote Login

1. **Local Login:** Whenever a user logs into its local system, it is known as local login.

The Procedure of Local Login

- Keystrokes are accepted by the terminal driver when the user types at the terminal.
- Terminal Driver passes these characters to OS.
- Now, OS validates the combination of characters and opens the required application.

Remote Login: Remote Login is a process in which users can log in to a remote site i.e. computer and use services that are available on the remote computer. With the help of remote login, a user is able to understand the result of transferring the result of processing from the remote computer to the local computer.

The Procedure of Remote Login

- When the user types something on the local computer, the local operating system accepts the character.
- The local computer does not interpret the characters, it will send them to the TELNET client.
- TELNET client transforms these characters to a universal character set called Network Virtual Terminal (NVT) characters and it will pass them to the local TCP/IP protocol Stack.
- Commands or text which are in the form of NVT, travel through the Internet and it will arrive at the TCP/IP stack at the remote computer.
- Characters are then delivered to the operating system and later on passed to the TELNET server.
- Then TELNET server changes those characters to characters that can be understandable by a remote computer.
- The remote operating system receives characters from a pseudo-terminal driver, which is a piece of software that pretends that characters are coming from a terminal.
- The operating system then passes the character to the appropriate application program.

TELNET Commands

Commands of Telnet are identified by a prefix character, Interpret As Command (IAC) with code 255. IAC is followed by command and option codes.

The basic format of the command is as shown in the following figure :

Advantages of Telnet

1. It provides remote access to someone's computer system.

2. Telnet allows the user for more access with fewer problems in data transmission.
3. Telnet saves a lot of time.
4. The oldest system can be connected to a newer system with telnet having different operating systems.

Disadvantages of Telnet

1. As it is somehow complex, it becomes difficult to beginners in understanding.
2. Data is sent here in form of plain text, that's why it is not so secured.
3. Some capabilities are disabled because of not proper interlinking of the remote and local devices.

Modes of Operation

Most telnet implementations operate in one of the following three modes:

1. Default mode
2. Character mode
3. Line mode

1. Default Mode: If no other modes are invoked then this mode is used. Echoing is performed in this mode by the client. In this mode, the user types a character and the client echoes the character on the screen but it does not send it until the whole line is completed.

2. Character Mode: Each character typed in this mode is sent by the client to the server. A server in this type of mode normally echoes characters back to be displayed on the client's screen.

3. Line Mode: Line editing like echoing, character erasing, etc. is done from the client side. The client will send the whole line to the server.

Firewall:

A firewall is a network security device, either hardware or software-based, which monitors all incoming and outgoing traffic and based on a defined set of security rules accepts, rejects, or drops that specific traffic.

- *Accept: allow the traffic*
- *Reject: block the traffic but reply with an "unreachable error"*
- *Drop : block the traffic with no reply*

A firewall is a type of network security device that filters incoming and outgoing network traffic with security policies that have previously been set up inside an organization. A firewall is essentially the wall that separates a private internal network from the open Internet at its very basic level.

Working of Firewall

Firewall match the network traffic against the rule set defined in its table. Once the rule is matched, associate action is applied to the network traffic. For example, Rules are defined as any employee from Human Resources department cannot access the data from code server and at the same time another rule is defined like system administrator can access the data from both Human Resource and technical department. Rules can be defined on the firewall based on the necessity and security policies of the organization. From the perspective of a server, network traffic can be either outgoing or incoming.

Firewall maintains a distinct set of rules for both the cases. Mostly the outgoing traffic, originated from the server itself, allowed to pass. Still, setting a rule on outgoing traffic is always better in order to achieve more security and prevent unwanted communication. Incoming traffic is treated differently. Most traffic which reaches on the firewall is one of these three major Transport Layer protocols- TCP, UDP or ICMP. All these types have a source address and destination address. Also, TCP and UDP have port numbers. ICMP uses *type code* instead of port number which identifies purpose of that packet.

1. Packet Filtering Firewall

Packet filtering firewall is used to control network access by monitoring outgoing and incoming packets and allowing them to pass or stop based on source and destination IP address, protocols, and ports. It analyses traffic at the transport protocol layer (but mainly uses first 3 layers). Packet firewalls treat each packet in isolation. They have no ability to tell whether a packet is part of an existing stream of traffic. Only It can allow or deny the packets based on unique packet headers. Packet filtering firewall maintains a filtering table that decides whether the packet will be forwarded or discarded.

- Incoming packets from network 192.168.21.0 are blocked.
- Incoming packets destined for the internal TELNET server (port 23) are blocked.
- Incoming packets destined for host 192.168.21.3 are blocked.
- All well-known services to the network 192.168.21.0 are allowed.

State ful Inspection Firewall

Stateful firewalls (performs Stateful Packet Inspection) are able to determine the connection state of packet, unlike Packet filtering firewall, which makes it more efficient. It keeps track of the state of networks connection travelling across it, such as TCP streams. So the filtering decisions would not only be based on defined rules, but also on packet's history in the state table.

3. **Software Firewall**

A software firewall is any firewall that is set up locally or on a cloud server. When it comes to controlling the inflow and outflow of data packets and limiting the number of networks that can be linked to a single device, they may be the most advantageous. But the problem with software firewall is they are time-consuming.

4. **Hardware Firewall**

They also go by the name "firewalls based on physical appliances." It guarantees that the malicious data is halted before it reaches the network endpoint that is in danger.

5. Application Layer Firewall

Application layer firewall can inspect and filter the packets on any OSI layer, up to the application layer. It has the ability to block specific content, also recognize when certain application and protocols (like HTTP, FTP) are being misused. In other words, Application layer firewalls are hosts that run proxy servers. A proxy firewall prevents the direct connection between either side of the firewall, each packet has to pass through the proxy.

6. Next Generation Firewalls (NGFW)

NGFW consists of Deep Packet Inspection, Application Inspection, SSL/SSH inspection and many functionalities to protect the network from these modern threats.

7. Proxy Service Firewall

This kind of firewall filters communications at the application layer, and protects the network. A proxy firewall acts as a gateway between two networks for a particular application.

8. **Circuit Level Gateway Firewall**

This works as the Sessions layer of the OSI Model's . This allows for the simultaneous setup of two Transmission Control Protocol (TCP) connections. It can effortlessly allow data packets to flow without using quite a lot of computing power. These firewalls are ineffective because they do not inspect data packets; if malware is found in a data packet, they will permit it to pass provided that TCP connections are established properly.

Advantages of using Firewall

- Protection from unauthorized access: Firewalls can be set up to restrict incoming traffic from particular IP addresses or networks, preventing hackers or other malicious actors from easily accessing a network or system. Protection from unwanted access.

- Prevention of malware and other threats: Malware and other threat prevention: Firewalls can be set up to block traffic linked to known malware or other security concerns, assisting in the defense against these kinds of attacks.
- Control of network access: By limiting access to specified individuals or groups for particular servers or applications, firewalls can be used to restrict access to particular network resources or services.
- Monitoring of network activity: Firewalls can be set up to record and keep track of all network activity.
- Regulation compliance: Many industries are bound by rules that demand the usage of firewalls or other security measures.
- Network segmentation: By using firewalls to split up a bigger network into smaller subnets, the attack surface is reduced and the security level is raised.

Disadvantages of using Firewall

- Complexity: Setting up and keeping up a firewall can be time-consuming and difficult, especially for bigger networks or companies with a wide variety of users and devices.
- Limited Visibility: Firewalls may not be able to identify or stop security risks that operate at other levels, such as the application or endpoint level, because they can only observe and manage traffic at the network level.
- False sense of security: Some businesses may place an excessive amount of reliance on their firewall and disregard other crucial security measures like endpoint security or intrusion detection systems.
- Limited adaptability: Because firewalls are frequently rule-based, they might not be able to respond to fresh security threats.
- Performance impact: Network performance can be significantly impacted by firewalls, particularly if they are set up to analyze or manage a lot of traffic.
- Limited scalability: Because firewalls are only able to secure one network, businesses that have several networks must deploy many firewalls, which can be expensive.
- Limited VPN support: Some firewalls might not allow complex VPN features like split tunneling, which could restrict the experience of a remote worker.
- Cost: Purchasing many devices or add-on features for a firewall system can be expensive, especially for businesses.

Bluetooth:

Bluetooth is a wireless LAN technology designed to connect devices of different functions such as telephones, notebooks, computers (desktop and laptop), cameras, printers, coffee makers, and so on. A Bluetooth LAN is an ad hoc network, which means that the network is formed spontaneously; the devices, sometimes called gadgets, find each other and make a network called a piconet. A Bluetooth LAN can even be connected to the Internet if one of the gadgets has this capability. A Bluetooth LAN, by nature, cannot be large. If there are many gadgets that try to connect, there is chaos.

Bluetooth technology has several applications. Peripheral devices such as a wireless mouse or keyboard can communicate with the computer through this technology. Monitoring devices can communicate with sensor devices in a small health care center. Home security devices

can use this technology to connect different sensors to the main security controller. Conference attendees can synchronize their laptop computers at a conference.

Bluetooth was originally started as a project by the Ericsson Company. It is named for Harald Blaatand, the king of Denmark (940-981) who united Denmark and Norway. *Blaatand* translates to *Bluetooth* in English.

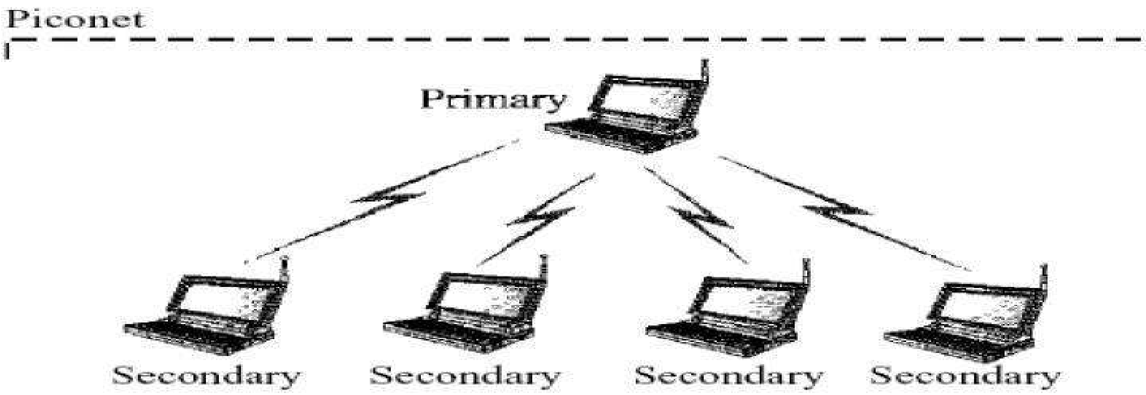
Today, Bluetooth technology is the implementation of a protocol defined by the IEEE 802.15 standard. The standard defines a wireless personal-area network (PAN) operable in an area the size of a room or a hall.

Architecture

Bluetooth defines two types of networks: piconet and scatternet.

Piconets

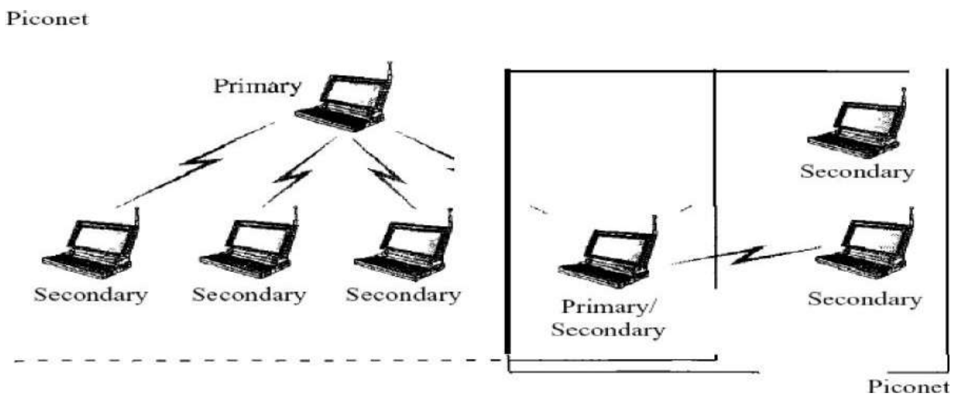
A Bluetooth network is called a piconet, or a small net. A piconet can have up to eight stations, one of which is called the primary's the rest are called secondary's. All the secondarystations synchronize their clocks and hopping sequence with the primary. Note that a piconet can have only one primary station. The communication between the primary and the secondary can be one-to-one or one-to-many. Figure 1 shows a piconet.



Although a piconet can have a maximum of seven secondary's, an additional eight secondary's can be in the *parked state.* A secondary in a parked state is synchronized with the primary, but cannot take part in communication until it is moved from the parked state. Because only eight stations can be active in a piconet, activating a station from the parked state means that an active station must go to the parked state.

Scatter net

Piconets can be combined to form what is called a scatter net. A secondary station in one piconet can be the primary in another piconet. This station can receive messages from the primary in the first piconet (as a secondary) and, acting as a primary, deliver them to secondary's in the second piconet. A station can be a member of two piconets.

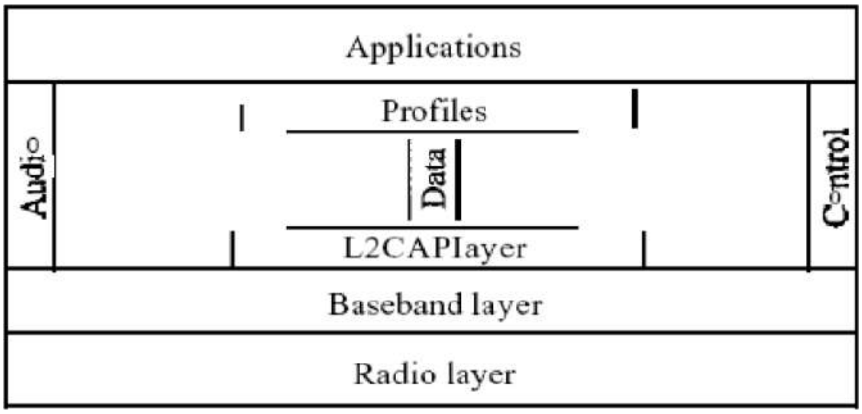


Bluetooth Devices

A Bluetooth device has a built-in short-range radio transmitter. The current data rate is 1 Mbps with a 2.4-GHz bandwidth. This means that there is a possibility of interference between the IEEE 802.11b wireless LANs and Bluetooth LANs.

Bluetooth Layers

Bluetooth uses several layers that do not exactly match those of the Internet model we have defined in this book. Figure 3 shows these layers.



Radio Layer

The radio layer is roughly equivalent to the physical layer of the Internet model. Bluetooth devices are low-power and have a range of 10 m.

Band

Bluetooth uses a 2.4-GHz ISM band divided into 79 channels of 1 MHz each.

FHSS

Bluetooth uses the frequency-hopping spread spectrum (FHSS) method in the physical layer to avoid interference from other devices or other networks. Bluetooth hops 1600 times per second, which means that each device changes its modulation frequency 1600 times per second. A device uses a frequency for only 625 Ils (1/1600 s) before it hops to another frequency; the dwell time is 625 *Ils.*

Modulation

To transform bits to a signal, Bluetooth uses a sophisticated version of FSK, called GFSK (FSK with Gaussian bandwidth filtering; a discussion of this topic is beyond the scope of this book). GFSK has a cannier frequency. Bit 1 is represented by a frequency deviationabove the carrier; bit as represented by a frequency deviation below the carrier. The frequencies, in megahertz, are defined according to the following formula for each channel: $fc=2402+n$ n =0, 1,2,3, ..., 78 For example, the first channel uses carrier frequency 2402 MHz (2.402 GHz), and the second channel uses carrier frequency 2403 MHz (2.403 GHz).

Baseband Layer

The baseband layer is roughly equivalent to the MAC sub layer in LANs. The access method is TDMA. The primary and secondary communicate with each other using time slots. The length of a time slot is exactly the same as the dwell time, 625 μs. This means that during the time that one frequency is used, a sender sends a frame to a secondary, or a secondary sends a frame to the primary. Note that the communication is only between the primary and a secondary; secondary’s cannot communicate directly with one another.