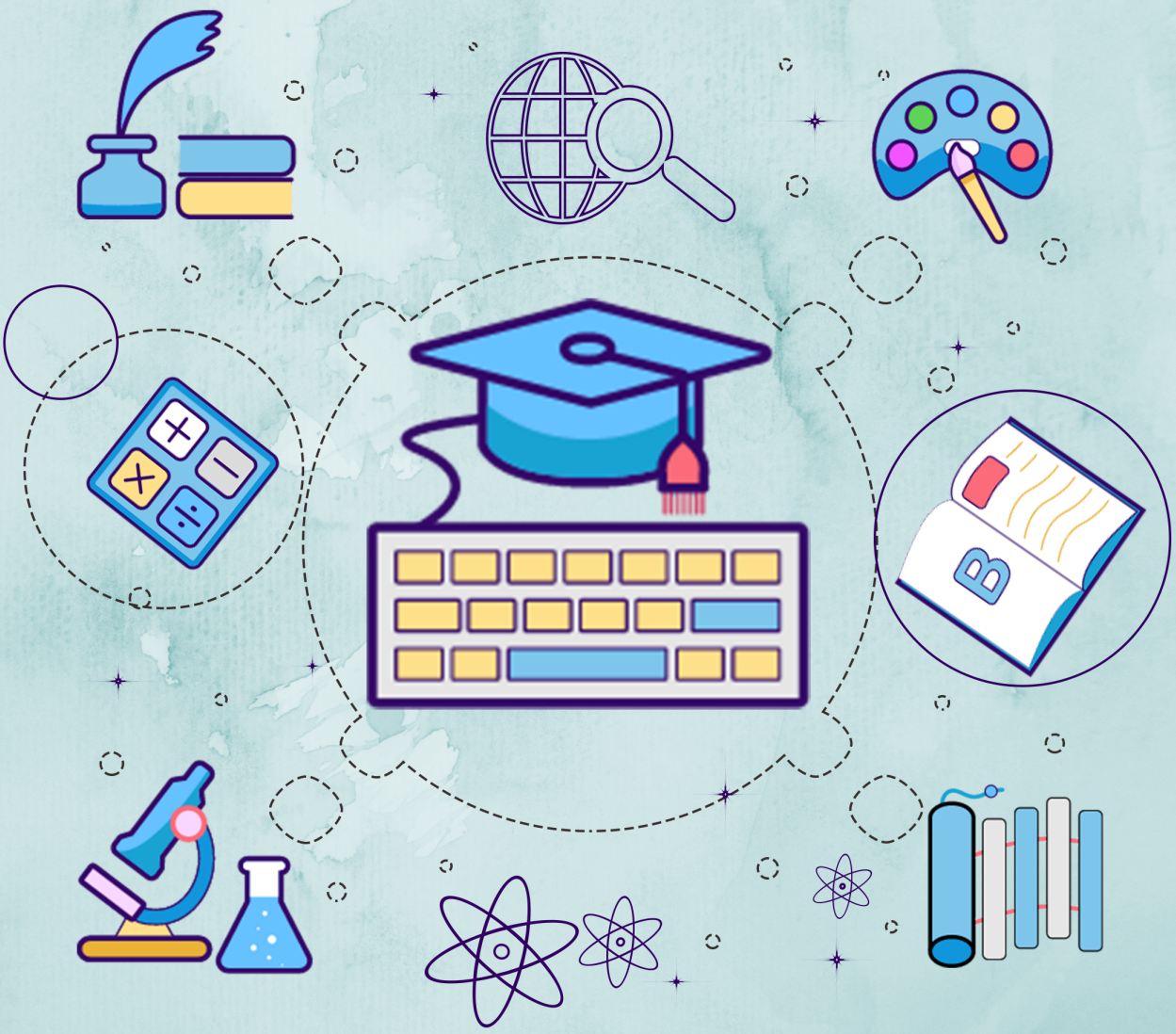
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

KeralaNotes



SYLLABUS | STUDY MATERIALS | TEXTBOOK

PDF | SOLVED QUESTION PAPERS

www.keralanotes.com

KeralaNotes

KTU STUDY MATERIALS

COMPUTER NETWORKS

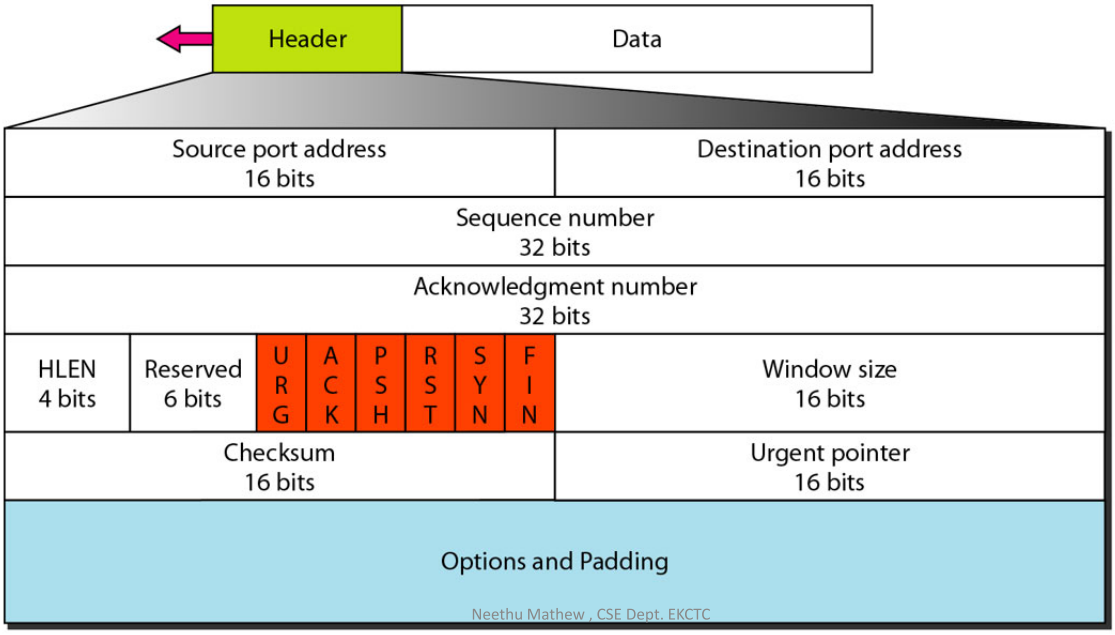
CST 303

Module 5

Related Link :

- KTU S5 STUDY MATERIALS
- KTU S5 NOTES
- KTU S5 SYLLABUS
- KTU S5 TEXTBOOK PDF
- KTU S5 PREVIOUS YEAR SOLVED QUESTION PAPER

TCP Header format



- ***Source port number (16 bits)*** – It defines the port number of the host that is sending the segment.(identifies the *TCP process* which sent the datagram.)
- ***Destination port number (16 bits)*** – It defines the port number of the host that is receiving the segment(field identifies the *TCP process* which is receiving the datagram.)
- ***Sequence number (32 bits)*** –identifies the first byte of the outgoing data. It defines the number assigned to the first byte of data contained in the segment. The receiver uses this to re-order segments arriving out of order and to compute an acknowledgement number.
- ***Acknowledgement number (32 bits)*** – Contains the next sequence number that the sender of the acknowledgement expects to receive which is the sequence number plus 1 (plus the number of bytes received in the last message). This field identifies the sequence number of the incoming data that is expected next.
- ***Header Length -HLEN***– This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes.

KTU S5 Computer Science

Computer Networks

Module 5

Module – 5 (Transport Layer and Application Layer)

Transport service – Services provided to the upper layers, Transport service primitives. User Datagram Protocol (UDP). Transmission Control Protocol (TCP) – Overview of TCP, TCP segment header, Connection establishment &release, Connection management modeling, TCP retransmission policy, TCP congestion control.

Application Layer –File Transfer Protocol (FTP), Domain Name System (DNS), Electronic mail, Multipurpose Internet Mail Extension (MIME), Simple Network Management Protocol (SNMP), World Wide Web(WWW) – Architectural overview.

Transport layer

- 4th layer of the OSI reference model.
- ultimate goal of the transport layer is to provide efficient, reliable, and cost-effective service to its users, normally processes in the application layer
- Transport layer is responsible for **process-to-process delivery** —the delivery of a packet, part of a message, from one process to another.
- Transport Layer Protocols **: TCP , UDP , SCTP**
- Functions/**services of Transport Layer**

1: Process-to- process communication

2: Port Addressing

3: Reliable Delivery

4: Flow Control , Error control

5: Multiplexing ,Demultiplexing

6: Encapsulation and Decapsulation

7: Connectionless vs ConnectionOriented

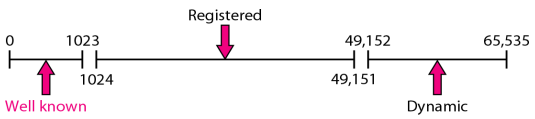
✓ **Addressing : Port numbers**

- Client/Server Paradigm

  Process on the local host, called Client, needs services from a process on the remote host called Server.
- At the **transport layer**, there is a **address** called **port number ,** to choose among the multiple processes running on the host.
- Transport layer address is specified with the help a 16-bit Port number in the **range of 0 and 65535.**

❑ Internet Assigned Number Authority (**IANA)** has divided the addresses in 3 ranges:

- Well Known ports (0 to 1023)
- Registered ports (1024 to 49,151)
- Dynamic ports (49,152 to 65,535)

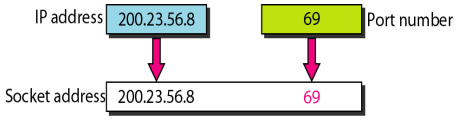


- **Well-known ports:** The ports in the range from 0 to 1023 are assigned and controlled by IANA. These are allocated to server services by IANA.
- **Registered ports(user ports):** Registered ports in the range from 1024 to 49151 are not assigned or controlled by IANA. These can be registered for services with the IANA and should be treated as semi-reserved.
- **Dynamic ports(private ports):** Dynamic ports (49152 to 65535) are neither controlled by IANA nor need to be registered. These are used by client programs and you are free to use these in client programs. Also known as **ephemeral ports.**

Well Known TCP Ports		Well Known UDP Ports:		Well Known TCP/UDP Common Ports:	
21	FTP	69	TFTP	53	DNS
23	Telnet	520	RIP	161	SNMP
25	SMTP				
80	HTTP				

Socket Address

- The combination of IP address and a port number is called a socket address
- The client socket address defines the client process uniquely just as the server socket address defines the server process uniquely.



Connectionless vs Connection Oriented

- ✓ **connection-oriented transport service**
- ✓ **connectionless transport service**

- **Connectionless**

  No need of connection establishment / no Connection release

  Packets not numbered; they may be delayed or lost or may arrive out of sequence.

  No acknowledgement mechanism

  Treats each segment as independent packet.

  UDP – User Datagram Protocol

- **Connection-Oriented**

  Connection is established / released

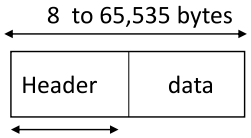
  Packets numbered

  Acknowledged

  TCP – Transmission Control Protocol

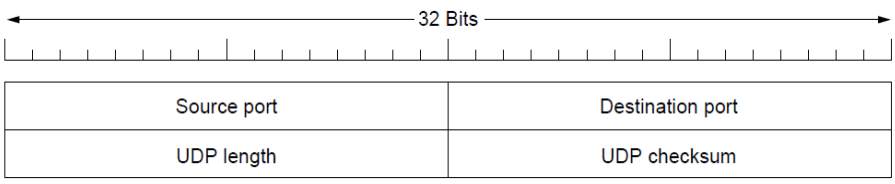
User Datagram Protocol : UDP

- Connectionless protocol
- Unreliable protocol
- Provide process-to-process communication
- No acknowledgement
- Overhead is minimum because of connectionless service
- Does not guarantee ordered delivery of data
- No Flow Control
- UDP is suitable for a process that requires simplest request- response communication with little concern for flow and error control.
- UDP packets are called user datagram
- UDP packets consist of a header and data



The UDP header format

8 bytes header consist of
- Source port number
- Destination port number
- UDP Length
- UDP Checksum



- **Source port (16 bits**): It defines the port number of the application program in the host of the sender.
- **Destination port (16 bits):** It defines the port number of the application program in the host of the receiver.
- **UDP Length:** specify entire length of UDP packet. It includes UDP Header information and data
- **UDP Checksum**: used for error control. A Checksum to verify that the end to end data has not been corrupted by routers or bridges in the network or by the processing in an end system. If this check is not required, the value of 0x0000 is placed in the field in which case the data is not checked by the receiver. It is optional, 0 in case it is not in use

Applications

- DNS (Domain Name Services)
- SNMP(Simple Network Management Protocol)
- routing information protocol RIP.
- Video streaming
- Client-Server Remote Procedure Call
- Real-time multimedia applications

TCP (Transmission Control Protocol)

- **Connection-oriented** protocol
- **Reliable** protocol
- Provide process-to- process communication
- **Stream oriented** protocol ,byte oriented
- Offers **full duplex communication** (segments move in both direction(2 way communication)
- Provides error checking and recovery mechanism
- Well known ports used by TCP : 80- HTTP ,25-SMTP, 23-TELNET, 53-DNS ,20 & 21 – FTP
- Provides flow control , congestion control and quality of service
- A Packet in TCP is called a **segment** (Packet consist of header and data)
- The segment consists of a 20 to 60 byte header, followed by data from the application program.
- The header is 20 bytes if there are no options and up to 60 bytes if it contains options

- ***Reserved*** – This is a 6-bit field reserved for future use.

Control flag bits

- ✓ **URG:** indicates **urgent pointer field** has significant data and should be processed.
- ✓ **ACK:** Indicates whether acknowledge field is valid.

  The **ACK bit** is set to **1** to indicate that the Acknowledgement number is valid.

  If **ACK is 0,** the segment does not contain an acknowledgement so the Acknowledgement number field is ignored.
- ✓ **PSH:** Push the data without buffering
- ✓ **RST:** Reset the connection
- ✓ **SYN:** Synchronize sequence numbers during connection establishment
- ✓ **FIN:** Terminate the connection

- ***Window Size*(16 bit)** – The *WINDOW* field identifies how much buffer space is available for incoming data.

During piggybacking, how much data a receiver is willing to accept.

Note: The process of sending data along with the acknowledgment is called piggybacking

- ***Checksum*(16 bit**) – used for error detection
- ***Urgent Pointer*** (16 bit) – This 16-bit field, which is valid only if the urgent flag is set, is used when the segment contains urgent data. It defines the number that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.
- ***Options*** – optional information in the TCP header. It provides a way to add extra facilities not covered by the regular header.

- TCP is A connection-oriented transport protocol
- It establishes a virtual path between the source and destination.
- All the segments belonging to a message are then sent over this virtual path.
- Using a single virtual pathway for the entire message facilitates the acknowledgement process as well as retransmission of damaged or lost frames. TCP, which uses the services of IP, a connection-less protocol, can be connection-oriented. TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself. If a segment is lost or corrupted, it is retransmitted.

Phases in TCP

- In TCP connection-oriented transmission requires **3 phases:**
  - Connection establishment
  - Data transfer
  - Connection termination

- A process running in one host(client) wants to initiate a connection with another process in another host(server), the client application process first informs the client TCP that it wants to establish a connection to a process in the server.

1. Connection establishment:

- TCP transmits data in full-duplex mode.

(TCP performs data communication in full-duplex mode, that is both the sender and receiver processes can send segments simultaneously.)

Three-way handshaking.

- **The connection establishment in TCP is called three way handshaking.**

- The TCP in the client then proceeds to establish a TCP connection with the TCP in the server in the following manner

Step1.

- The client-side TCP first sends a special TCP segment to the server-side TCP. This special segment contains no application-layer data.
- But one of the flag bits in the segment's header, the SYN bit, is set to 1.
- For this reason, this special segment is referred to as a SYN segment.
- In addition, the client randomly chooses an initial sequence number (client_isn) and puts this number in the sequence number field of the initial TCP SYN segment.
- This segment is encapsulated within an IP datagram and sent to the server.

Step2.

- Once the IP datagram containing the TCP SYN segment arrives at the server host, the server extracts the TCP SYN segment from the datagram, allocates the TCP buffers and variables to the connection, and sends a connection-granted segment to the client TCP.

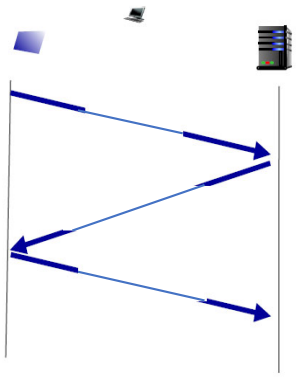
- This connection-granted segment also contains no application layer data.
- However, it does contain three important pieces of information in the segment header.
  - First, the SYN bit is set to 1.
  - Second, the acknowledgment field of the TCP segment header is set to client_isn+1.
  - Finally, the server chooses its own initial sequence number (server_isn) and puts this value in the sequence number field of the TCP segment header.
- The connection granted segment is referred to as a **SYN ACK segment**.

Note:

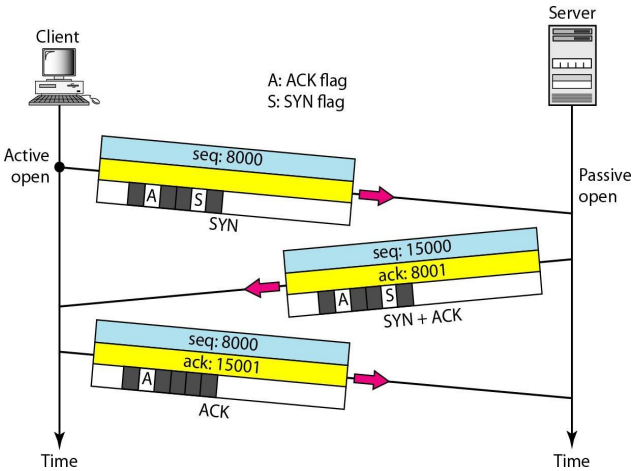
- This connection-granted segment is saying, in effect, "I received your SYN packet to start a connection with your initial sequence number, client_isn. I agree to establish this connection.
- My own initial sequence number is server_isn."

Step 3

- Upon receiving the SYN ACK segment, the client also allocates buffers and variables to the connection.
- The client host then sends the server yet another segment; this last segment acknowledges the server's connection-granted segment (the client does so by putting the value server_isn+1 in the acknowledgment field of the TCP segment header).
- The SYN bit is set to zero, since the connection is established.
- This third stage of the three-way handshake may carry client-to-server data in the segment payload
- *This connection establishment procedure is often referred to as a* ***three-way handshake***



TCP three-way handshake: segment exchange



2. Data Transfer

- After connection is established, bidirectional data transfer can take place.
- The client and server can both send data and acknowledgements.
- This data transfer process includes two process like pushing data and Urgent data.

3. Connection Termination

- Any of the two parties involved in exchanging data (client or server) can close the connection, although it is usually initiated by the client.

	TCP	UDP
Acronym for	Transmission Control Protocol	User Datagram Protocol or Universal Datagram Protocol
Connection	TCP is a connection-oriented protocol.	UDP is a connectionless protocol
Function	As a message makes its way across the internet from one computer to another. This is connection based.	UDP is also a protocol used in message transport or transfer. This is not connection based which means that one program can send a load of packets to another and that would be the end of the relationship.
Usage .	TCP is suited for applications that require high reliability, and transmission time is relatively less critical.	UDP is suitable for applications that need fast, efficient transmission, such as games. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients
Use by other protocols	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, TFTP, SNMP, RIP, VOIP.
Ordering of data packets	Segment sequencing. TCP rearranges data packets in the order specified	No Segment sequencing. UDP has no inherent order as all packets are independent of each other. If ordering is required, it has to be managed by the application layer.
Speed of transfer	The speed for TCP is slower than UDP.	UDP is faster because error recovery is not attempted. It is a "best effort" protocol.
Reliability	Reliable, There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent.	Unreliable ,There is no guarantee that the messages or packets sent would reach at all.

Header Size Common	TCP header size is 20 bytes	UDP Header size is 8 bytes
Header Fields	Source port, Destination port, Check Sum	Source port, Destination port, Check Sum
Streaming of data	Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries	Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent.
Weight	TCP is heavy-weight. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP is lightweight. There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.
Data Flow Control	TCP does Flow Control. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP does not have an option for flow control
Error Checking .	TCP does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.	UDP does error checking but simply discards erroneous packets. Error recovery is not attempted
Fields	1. Sequence Number, 2. AcK number, 3. Data offset, 4. Reserved, 5. Control bit, 6. Window, 7. Urgent Pointer 8. Options, 9. Padding, 10. Check Sum, 11. Source port, 12. Destination port	1. Length, 2. Source port, 3. Destination port, 4. Check Sum
Acknowledgement	Acknowledgement segments. Acknowledge sequencing.	No Acknowledgment No Acknowledge sequencing
Handshake	SYN, SYN-ACK, ACK	No handshake (connectionless protocol)

- **Transport Service Primitives**

purpose of the transport layer is to provide a reliable service on top of an unreliable network

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

Fig: The primitives for a simple transport service

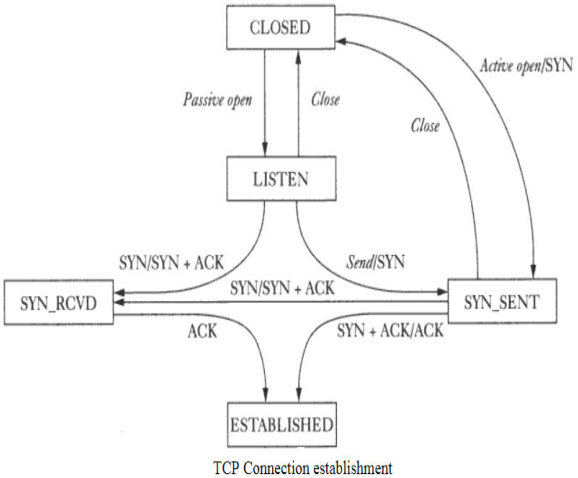
socket primitives (Berkeley sockets) used for TCP

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

TCP connection management modeling

State transition diagram

- The state diagram approach is to view the TCP connection establishment and closing simplifies the design of TCP implementation.
- The idea is to represent the TCP connection state, which progresses from one state to other as various messages are exchanged.
- To simplify the matter, we considered two state diagrams, viz., for TCP connection establishment and TCP connection closing.
- Fig 1 shows the state diagram for the TCP connection establishment and associated table briefly explains each state.



TCP Connection establishment

Listen

- This specifically applies to a Server.
- Represents the state when waiting for connection request from any remote host and port.
- From this state, the server can close the service or actively open a connection by sending SYN.

Syn-Sent

- This applies to both server and client side.
- Represents waiting for a matching of a connection request after having sent a connection request.
- Even though server is considered as the one with passive open, it can also send a SYN packet actively.

Syn-Rcvd

- Represents waiting for a confirmation connection request acknowledgment after having both received and sent connection request.

Established :

- Represents an open connection. Data transfer can take place from this point onwards

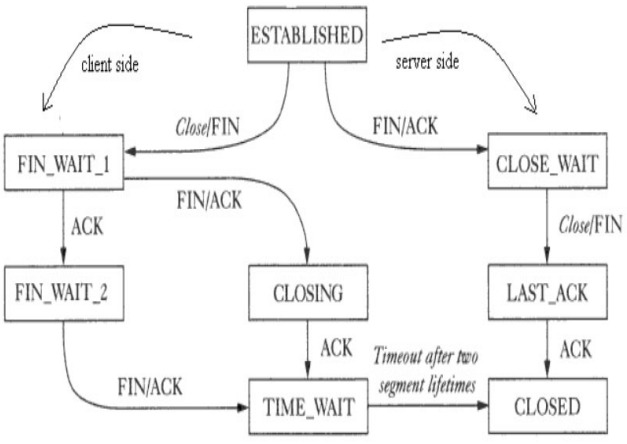


Fig 2. TCP Connection termination

FIN-WAIT-1

- Represents connection termination request from the remote TCP peer, or an acknowledgment of the connection termination request previously sent.
- This state is entered when server issues close call.

FIN-WAIT-2

- Represents waiting for a connection termination request from the remote TCP.

CLOSING

- Represents connection termination request **acknowledgment** from the remote TCP.

TIME_WAIT

- This represents waiting time enough for the packets to reach their **destination**.
- This waiting time is usually **4 min**.

CLOSE_WAIT

- Represents a state when the server receives a FIN from the remote TCP, sends ACK and issues **close** call sending FIN

LAST_ACK

- Represents waiting for an ACK for the previously sent FIN-ACK to the remote TCP

CLOSE

- Represents a closed TCP connection having received all the ACKs

Application Layer

- ✓ Provide services to the user
- ✓ It is the layer through which users interact.
- ✓ Specific services provided:
  - **File transfer, access, and management**: This application allows a user to access files (to make changes or read data), to retrieve files , and to manage or control files.
  - **Mail services:** This application provides the basis for e-mail forwarding and storage
  - **Directory services**: This application provides distributed database sources and access for global information about various objects and services.
- ✓ Protocols used : FTP, SMTP, HTTP etc

File Transfer Protocol (FTP)

- application layer protocol
- copying/transferring a file from one host to another.
- It establishes two connections between the hosts.
- To transfer a file, 2 TCP connections are used by FTP in parallel: **control connection & data connection**
- 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.
- FTP uses the services of TCP
- FTP uses two well-known TCP ports: Port 21 is used for the control connection, and port 20 is used for the data connection.

Neethu Mathew , CSE Dept. EKCTC

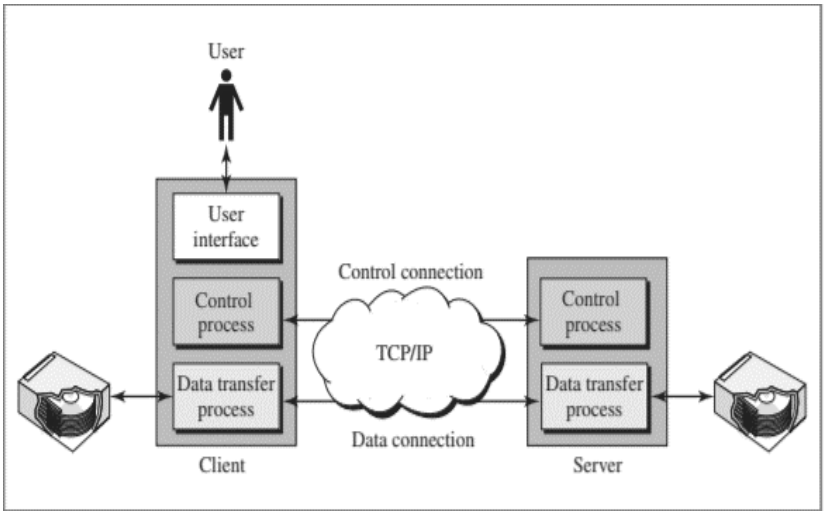


Figure :Basic model of FTP

Neethu Mathew , CSE Dept. EKCTC

- Figure shows the basic model of FTP
- The client has 3 components: the user interface, the client control process, and the client data transfer process.
- The server has 2 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.
- 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.

FTP Transmission modes

FTP can transfer a file across the data connection by using one of the following three transmission modes:

- stream mode :It is the default mode. Data are delivered as a continuous stream of bytes.
- block mode :Data can be delivered in blocks.
- compressed mode : the data can be compressed

File Type

FTP can transfer one of the following file types across the data connection:

- ASCII file
- EBCDIC file
- image file

12.11.1. Communication in FTP

FTP operates in client – server environment. The two computers involved in communication may be different in terms of the operating systems, character sets, file structures and file formats. FTP make them compatible. The approaches for communication over control connection and data connection are different from each other.

1. Communication Over Control Connection

Let us consider figure 12.17 to understand the FTP's approach for the communication over the control connection.

Similar to SMTP, FTP uses a set of ASCII characters to communicate across the control connection. Communication is achieved through commands and response. One command is sent at a time. Each command or response is only of one short line. Therefore, it is not necessary to think about file structure. Each line is terminated with a two character end of line token such as carriage return and line feed.

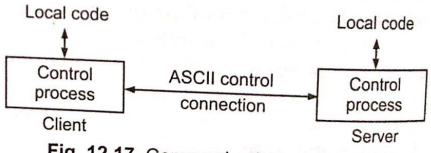


Fig. 12.17. Communication over control connection

2. Communication Over Data Connection

The purpose of implementing a data connection is to transfer a file. For this, the client has to define the following :

(i) Type of file being transferred.
(ii) Structure of data and
(iii) Transmission mode.

Before the transmission over data connection, the communication over control connection is performed. Let us consider figure 12.18 to understand communication over data connection. The problem of heterogeneity is solved by defining three attributes of communication : file type, data structure and transmission mode. Let us discuss them one by one.

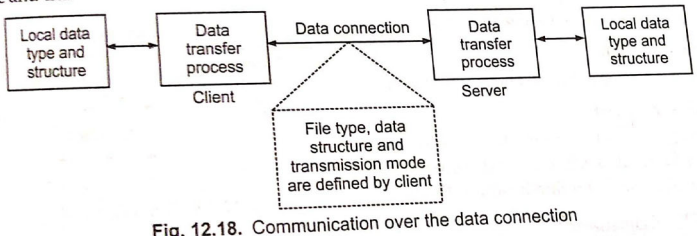


Fig. 12.18. Communication over the data connection

Domain Name System (DNS)

Why DNS is Needed?

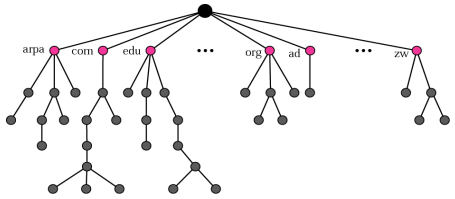
- To identify an entity, TCP/IP protocols use the IP address, which uniquely identifies the connection of a host to the Internet.
- However, people prefer to use names instead of numeric addresses.
- Therefore, we need a system that can map a name to an address or an address to a name.

- DNS is the naming service of the internet that translate host names to IP addresses and vice versa
- Host name are structured character strings eg.www.google.com.
- IP addresses are 32- bit integers eg. 139.130.4.5

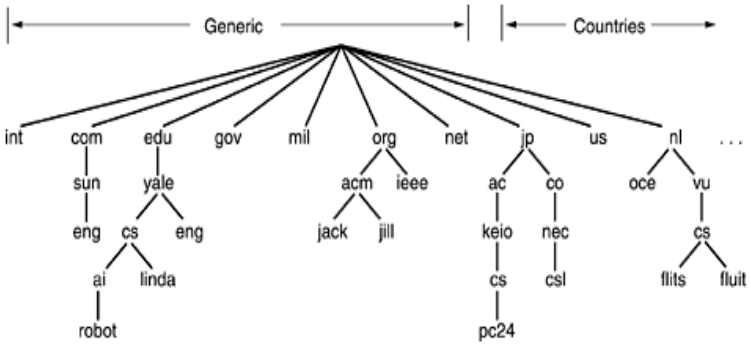
Name Spaces

- The names assigned to machines must be unique because the addresses are unique.
- A name space that maps each address to a unique name can be organized in two ways:
  - ❑ Flat Name Space
  - ❑ Hierarchical Name Space
- In Flat Name Space , a name in this space is a Sequence of characters without structure. A name is assigned to an address. The main disadvantage of flat namespace is that, it cannot use in a large system such as the internet.
- In Hierarchical Name Space, Each name has several parts. The first part can defined the nature of the organization, the second part can defined the name, and the third part can define department and so on.

Domain Name Space

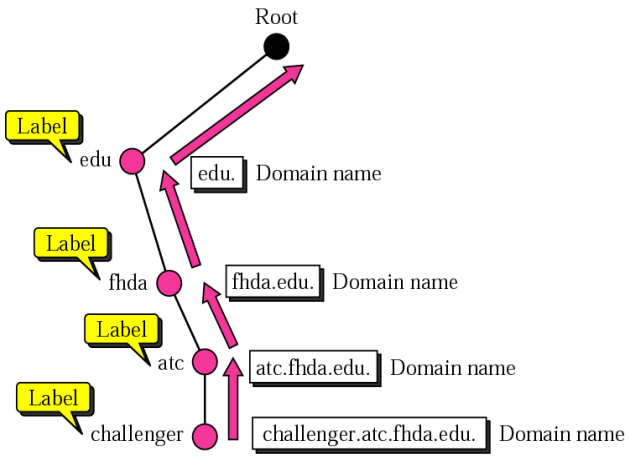


- The domain name space is hierarchical in design.
- The names are defined in an inverted-tree structure with the root at the top.
- Conceptually, the Internet is divided into over 200 top-level domains, where each domain covers many hosts.
- Each domain is partitioned into subdomains, and these are further partitioned, and so on.
- All these domains can be represented by a tree, The leaves of the tree represent domains that have no subdomains.



Domain names and labels

- Each node in the tree has a label which is a string with a maximum of 63 characters
- 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.
- Example:- challenger.atc.fhda.edu. is a domain name that defines a computer at De Anza College.
- Each label in the domain may define an entity in the organization; the level of detail increases from right to left.



Fully Qualified Domain Name (FQDN)

- 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.
- It contains all labels, from the most specific to the most general, that uniquely define the name of the host.
- Ex: **challenger.ate.tbda.edu.**

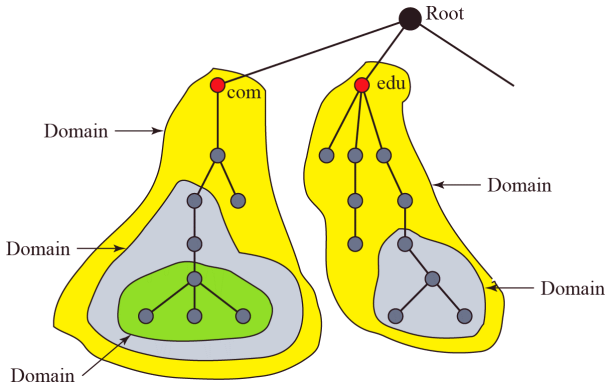
Partially Qualified Domain Name (PQDN)

- 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.
- Ex: **challenger.atc.fhda.edu**

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

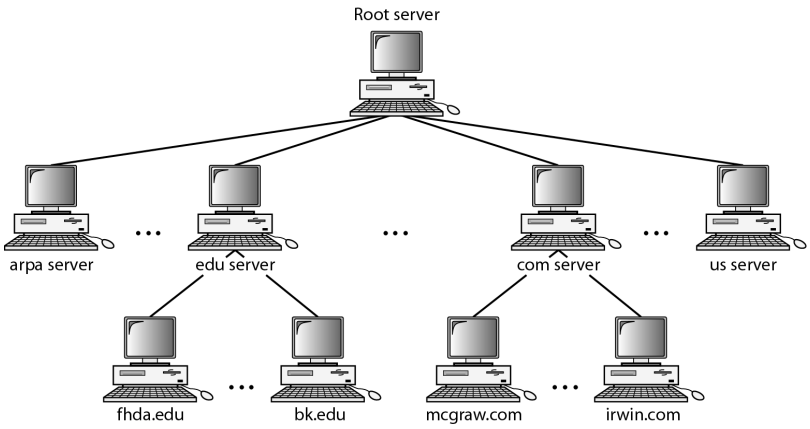
Domains

- A domain is a subtree of the domain space.
- The name of the domain is the name of the node at the top of the subtree.
- Note that a domain may itself be divided into domains (subdomains)



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. The solution to these problems is to distribute the information among many computers called DNS servers.

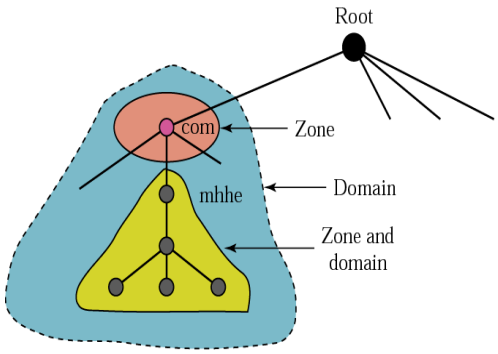


DNS servers

- 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.
- 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.
- ie, we have a hierarchy of servers in the same way that we have a hierarchy of names.

Zone

- What a server is responsible for or has authority over is called a zone.
- It can be defined as a contiguous part of the entire tree.
- 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.
- The information about the nodes in the subdomains is stored in the servers at the lower levels.
- 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



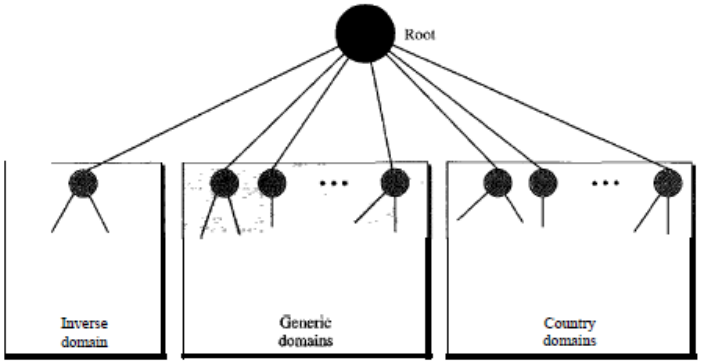
Root server

- 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.
- 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 updating is required, it must be done by the primary server, which sends the updated version to the secondary.

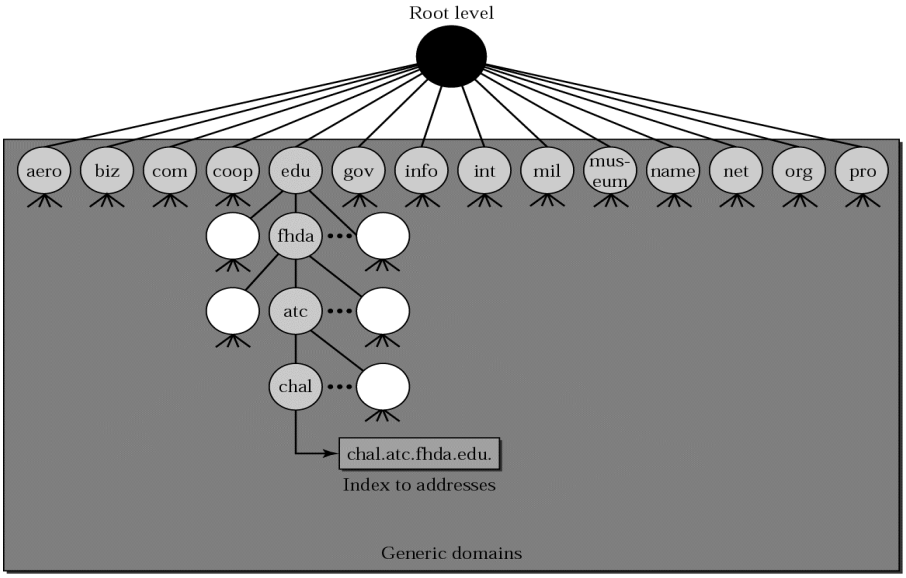
DNS in the internet

In the Internet, the domain name space (tree) is divided into three different sections:

- generic domains,
- country domains, and
- inverse domain



❑ **Generic domains** define registered hosts according to their generic behaviour. Each node in the tree defines a domain, which is an index to the domain name space database.



Generic domain labels

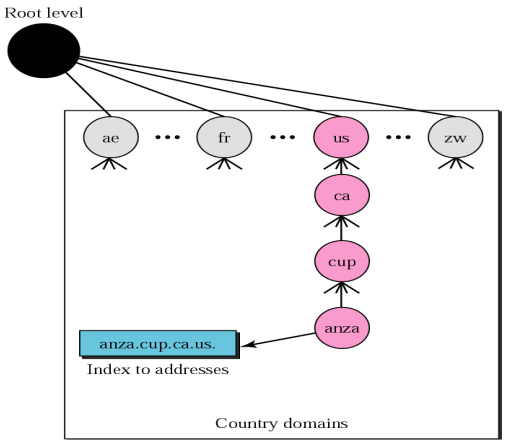
Label	*Description*
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to "com")
com	Commercial organizations
coop	Cooperative business organizations
edu	Educational institutions
gov	Government institutions
info	Information service providers

Label	*Description*
int	International organizations
mil	Military groups
museum	Museums and other non-profit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

❑ **Country domains**

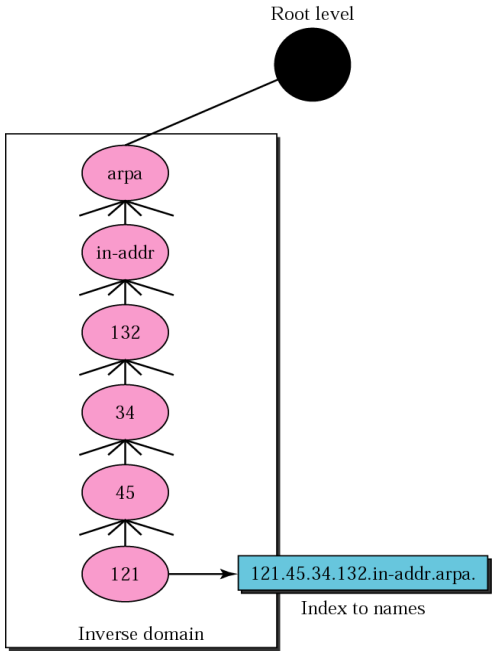
- This section uses two-character country abbreviations (e.g., us for United States). Second labels can be organizational, or they can be more specific, national designations. The United States, for example, uses state abbreviations as a subdivision of us (e.g., ca.us.).

Domain Name	Meaning
au	Australia
in	India
cl	Chile
fr	France
us	United States
za	South Africa
uk	United Kingdom
jp	Japan
es	Spain
de	Germany
ca	Canada
ee	Estonia
hk	Hong Kong



❑ Inverse domain

- The inverse domain is used to map an address to a name.



Resolution

- Mapping a name to an address or an address to a name is called name-address resolution.

Resolver

- 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.

Mapping Names to Addresses

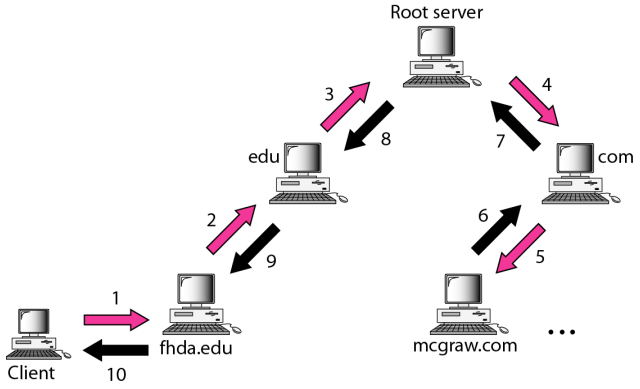
Most of the time, the resolver gives a domain name to the server and asks for the corresponding address. In this case, the server checks the generic domains or the country domains to find the mapping. If the domain name is from the generic domains section, the resolver receives a domain name such as "chal.atc.jhda.edu.". The query is sent by the resolver to the local DNS server for resolution. If the local server cannot resolve the query, it either refers the resolver to other servers or asks other servers directly. If the domain name is from the country domains section, the resolver receives a domain name such as "ch.jhda.cu.ca.us.". The procedure is the same.

Mapping Addresses to Names

A client can send an IP address to a server to be mapped to a domain name. As mentioned before, this is called a PTR query. To answer queries of this kind, DNS uses the inverse domain. However, in the request, the IP address is reversed and the two labels in-addr and arpa are appended to create a domain acceptable by the inverse domain section. For example, if the resolver receives the IF address 132.34.45.121, the resolver first inverts the address and then adds the two labels before sending. The domain name sent is "121.45.34.132.in-addr.arpa." which is received by the local DNS and resolved.

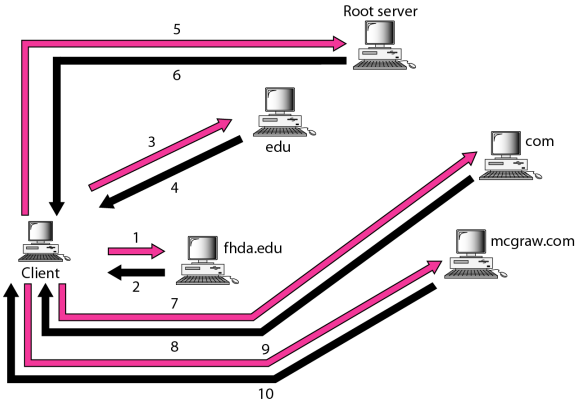
- **Recursive Resolution**

The client (resolver) can ask for a recursive answer from a name server. This means that the resolver expects the server to supply the final answer. If the server is the authority for the domain name, it checks its database and responds. If the server is not the authority, it sends the request to another server (the parent usually) and waits for the response. If the parent is the authority, it responds; otherwise, it sends the query to yet another server. When the query is finally resolved, the response travels back until it finally reaches the requesting client. This is called recursive resolution and is shown in Figure



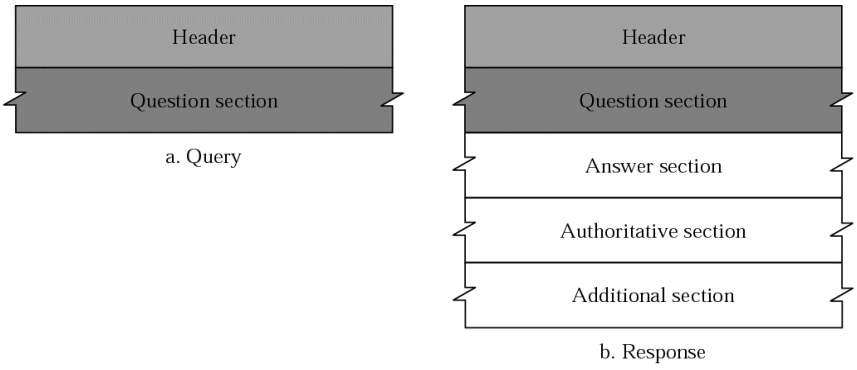
- **Iterative Resolution**

If the client does not ask for a recursive answer, the mapping can be done iteratively. If the server is an authority for the name, it sends the answer. If it is not, it returns (to the client) the IP address of the server that it thinks can resolve the query. The client is responsible for repeating the query to this second server. If the newly addressed server can resolve the problem, it answers the query with the IP address; otherwise, it returns the IP address of a new server to the client. Now the client must repeat the query to the third server. This process is called iterative resolution because the client repeats the same query to multiple servers. In Figure the client queries four servers before it gets an answer from the mcgraw.com server.



DNS messages

- 2 types of messages: **query** and **response**
- The query message consists of a header and question records;
- the response message consists of a header, question records, answer records, authoritative records, and additional records



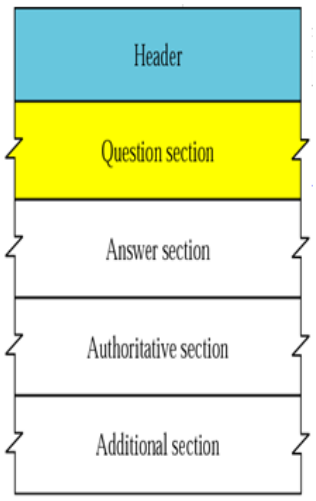
Header format

- **Identification**: 2 byte field, used by the client to match the response with the query. The client uses a different identification number each time it sends a query..
- **Flags**:

  1.QR Query/Response: One bit, 0=query 1=response.
  2.Opcode: four bits define type of query or response 0=normal, 1=inverse, 2=server status is requested.
  3. AA authoritative answer: One bit , server responding is authoritative server. response message.
  4. TC truncated: One bit, if it equals 1 means answer was larger than 512 bytes and was truncated.
  5. RD recursion desired: One bit, if set to 1 means client wants a recursive answer.
  6. RA recursion available: One bit, when set to 1 means a recursive response is available. response message.
  7. Reserved: three bit field set to 000
  8. rCode: Four bit field contains error status.
- **Number of question records**: subfield contains the number of queries in the question section of the message.
- **Number of answer records:** subfield contains the number of answer records in the answer section of the response message.
- **Number of Authoritative Records:** Two byte field, number of authoritative records in the authoritative section of a response message
- **Number of Additional Records**: Two byte field, the number additional records in the additional section of a response message.

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)

Rest of the DNS Message



- Question Section: Section consisting of one or more question records. Exists in both query and response
- Answer Section: Section consisting of one or more resource records. Exists in response only. This section includes the answer from the server to the client (resolver).
- Authoritative Section: Section consisting of one or more resource records. Exists in response only. This contains the domain name about one or more of the authoritative servers for the query.
- Additional Info Section: Contains one or more resource records. Exists in response only. Eg.,IP address of authoritative server.

Electronic mail *(e-mail)*

- One of the most popular Internet services is electronic mail (e-mail)
- 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.
- Email messages are comprised of three components:
  - ✓ Message envelop: Describes the email's electronic format
  - ✓ Message header: Includes sender/recipient information and email subject line
  - ✓ Message body: Include text, image and file attachments

Basic email functions :

- ❑ **Composition** : refers to the process of creating messages and answers
- ❑ **Transfer** : refers to moving messages from the sender to the recipient
- ❑ **Reporting** :telling the sender what happened to the message. Was it delivered? Was it rejected? Was it lost? Numerous applications exist in which confirmation of delivery is important.
- ❑ **Displaying** : incoming messages is needed so people can read their e-mail
- ❑ **Disposition** : what the recipient does with the message after receiving it. Possibilities include throwing it away before reading, throwing it away after reading, saving it, forwarding and so on.

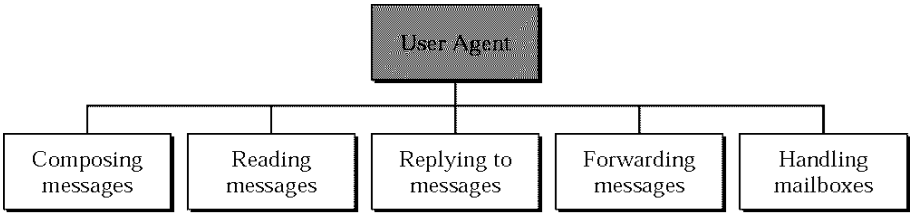
Electronic mail (e-mail)

The sending of electronic mail in the Internet requires these components:

- ***user agents (UA),***
- ***Mail transfer agents (MTA)***
- ***The protocol that controls mail delivery***

User agent (UA)

- It provides service to the user to make the process of sending and receiving a message easier.



- UA : a program that accepts a variety of commands for composing, receiving, and replying to messages, as well as for manipulating mailboxes.
- The user agent is not responsible for sending or receiving email.
- There are two types of user agents: command-driven and GUI-based

Mail Transfer Agent (MTA)

- The actual mail transfer requires a mail transfer agent (MTA).
- To send mail, a system must have a client MTA, and to receive mail, a system must have a server MTA.

Electronic mail - Architecture

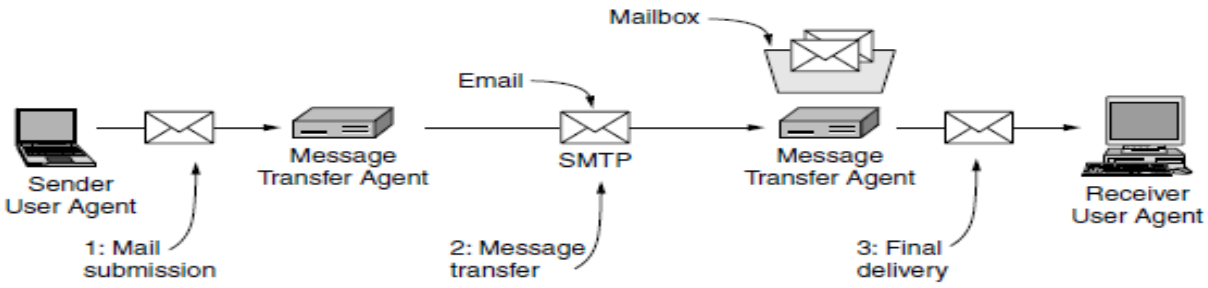
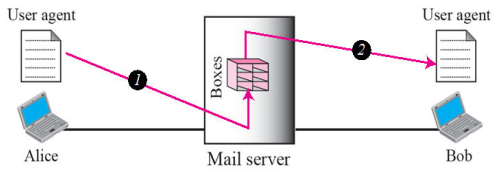


Figure 7-7. Architecture of the email system.

- To explain the architecture of email, we give 4 scenarios.
- The **fourth scenario is the most common** in the exchange of email.

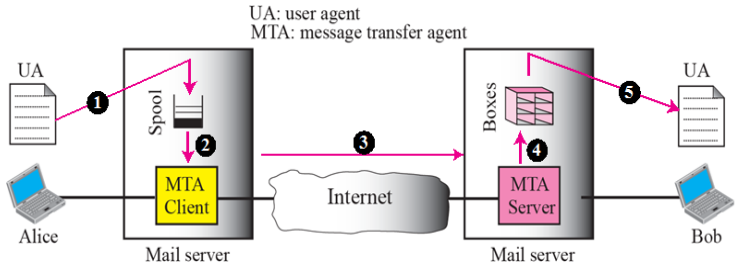
First scenario

- When the sender and the receiver of an email are on the same system, we need only two user agents.



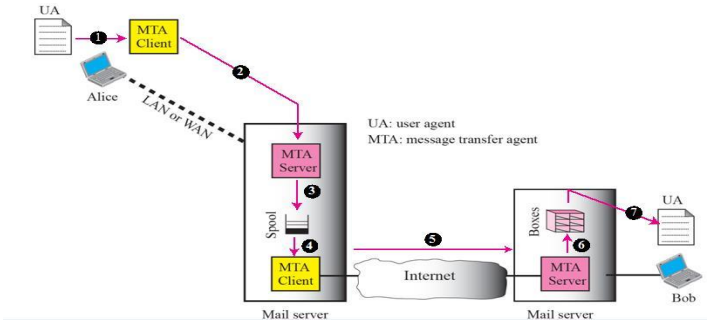
Second scenario

- When the sender and the receiver of an email are on different systems, we need two UAs and a pair of MTAs (client and server).



Third scenario

- When the sender is connected to the mail server via a LAN or a WAN, we need two UAs and two pairs of MTAs (client and server).



Fourth scenario

- When both sender and receiver are connected to the mail server via a LAN or a WAN, we need two UAs, two pairs of MTAs (client and server), and a pair of Message Access Agent -MAAs (client and server).
- This is the most common situation today.

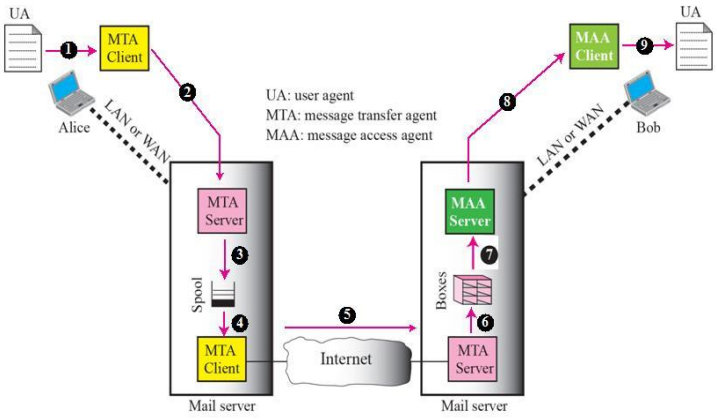
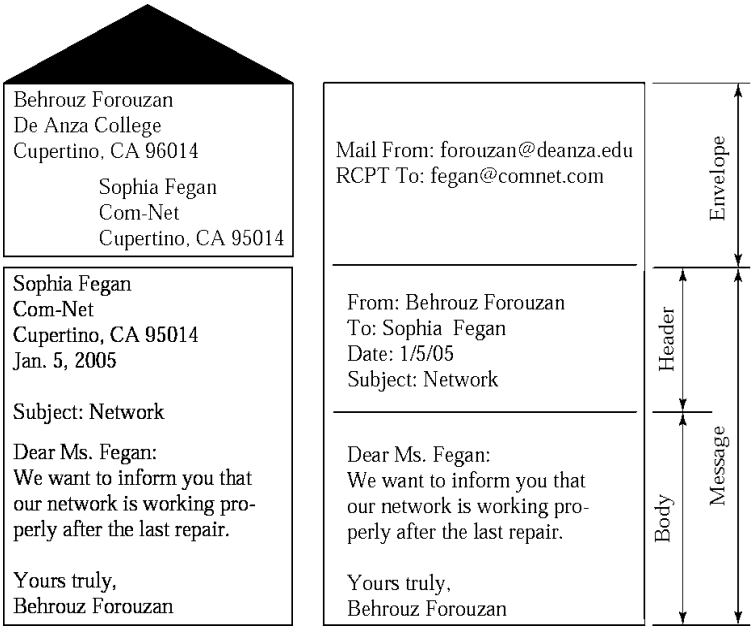


Figure : *Format of an email*



Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	E-mail address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User-chosen keywords
Subject:	Short summary of the message for the one-line display

MIME (Multi - Purpose Internet Mail Extensions)

- Electronic mail has a simple structure. It can send messages only in NVT 7-bit ASCII format. In other words, it has some limitations. For example, it cannot be used for languages that are not supported by 7-bit ASCII characters (such as French, German, Hebrew, Russian, Chinese, and Japanese). Also, it cannot be used to send binary files or video or audio data.
- Multipurpose Internet Mail Extensions (MIME) is **a supplementary protocol** that allows **non-ASCII data** to be sent through e-mail.
- MIME **transforms non-ASCII data at the sender site to NVT ASCII data** and delivers them to the client MTA to be sent through the Internet. The message at the receiving side is transformed back to the original data.
- We can think of MIME as a set of software functions that transforms non-ASCII data (stream of bits) to ASCII data and vice versa, as shown in Figure

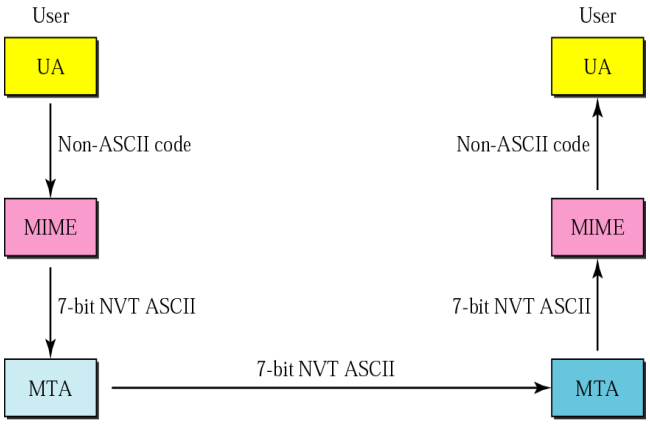


Figure : *MIME*

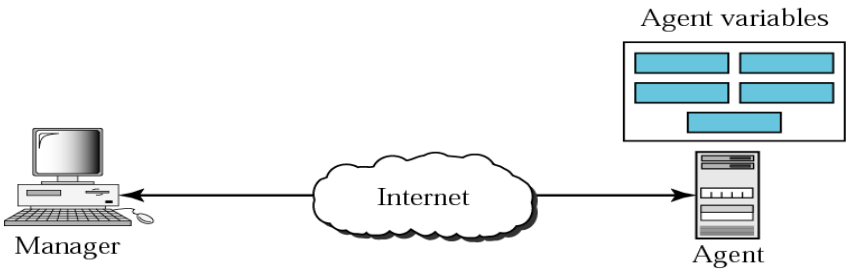
- MIME defines 5 headers that can be added to the original e-mail header section to define the transformation parameters:

1. MIME-Version
2. Content-Type
3. Content-Transfer-Encoding
4. Content-Id
5. Content-Description

Email header	
MIME-Version: 1.1 Content-Type: type/subtype Content-Transfer-Encoding: encoding type Content-Id: message id Content-Description: textual explanation of nontextual contents	MIME headers
Email body	

SNMP (Simple Network Management Protocol)

- 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.
- That is, a manager, usually a host, controls and monitors a set of agents, usually routers



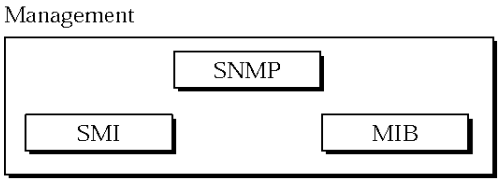
- SNMP is an application-level protocol in which a few manager stations control a set of agents.
- It can be used in a heterogeneous network

- 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.

- 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**



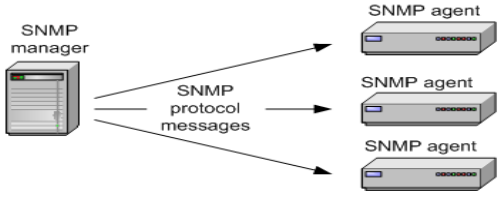
- To do management tasks, SNMP uses two other protocols:
  - ❑ Structure of Management Information (SMI)
  - ❑ Management Information Base (MIB).
- Management on the Internet is done through the cooperation of the three protocols SNMP, SMI, and MIB
- SMI:- defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values
- MIB:- For each entity to be managed, this protocol must define the number of objects, name them according to the rules defined by SMI, and associate a type to each named object.

✓ **Role of SNMP**

- It defines the format of the packet to be sent from a manager to an agent and vice versa.
- It also interprets the result and creates statistics (often with the help of other management software).
- The packets exchanged contain the object (variable) names and their status (values).
- SNMP is responsible for reading and changing these values.

In SNMP : -

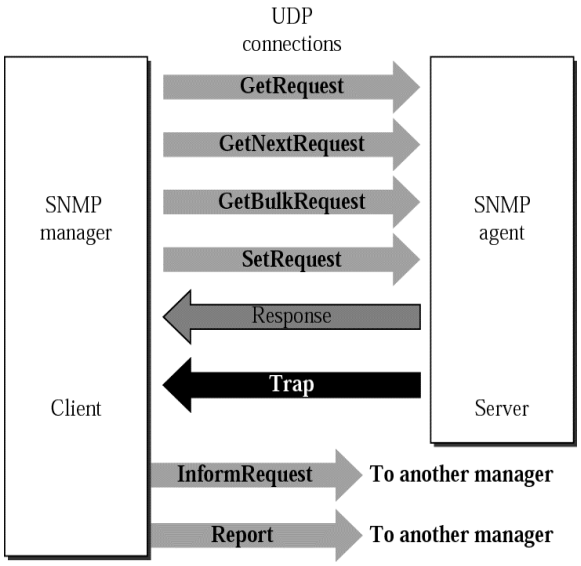
- SNMP agent
- SNMP manager
- Managed devices
- Management Information Bases (MIBs)



SNMP agent is software that runs on a piece of network equipment (host, router, printer, or others) and that maintains information about its configuration and current state in a database. Information in the database is described by Management Information Bases (MIBs). An SNMP manager is an application program that contacts an SNMP agent to query or modify the database at the agent. SNMP protocol is the application layer protocol used by SNMP agents and managers to send and receive data.

SNMP PDU (protocol data units)

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



- **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
- **GetBulkRequest:** The GetBulkRequest POD is sent from the manager to the agent to retrieve a large amount of data.
- **SetRequest:** The SetRequest PDD is sent from the manager to the agent to set (store) a value in a variable

- **Response :** 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:** Trap is sent from the agent to the manager to report an event.
- **InformRequest:** sent from one manager to another remote manager to get the value of some variables from agents under the control of the remote manager.
- **Report :**to report some types of errors between managers.

❑ A message in SNMPv3 is made of four elements: version, header, security parameter and data.

World Wide Web (WWW)

- www/web/w3
- WWW is a repository of information linked together from points all over the world
- Interconnected system of public webpages accessible through internet
- A technical definition of the World Wide Web is : all the resources and users on the Internet that are using the Hypertext Transfer Protocol (HTTP).

Architecture

- WWW is a distributed client server 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
- 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.

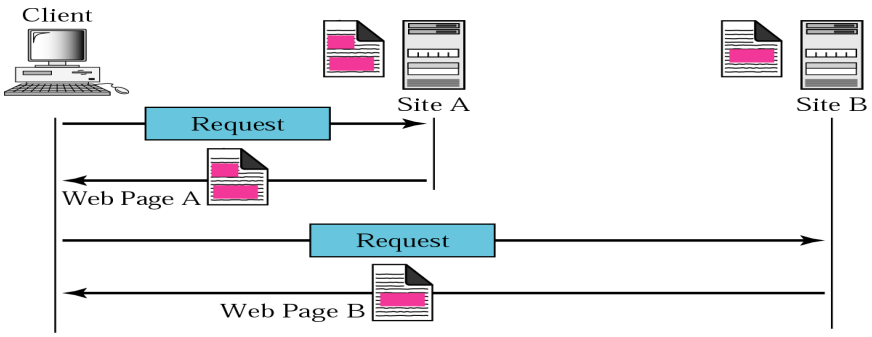
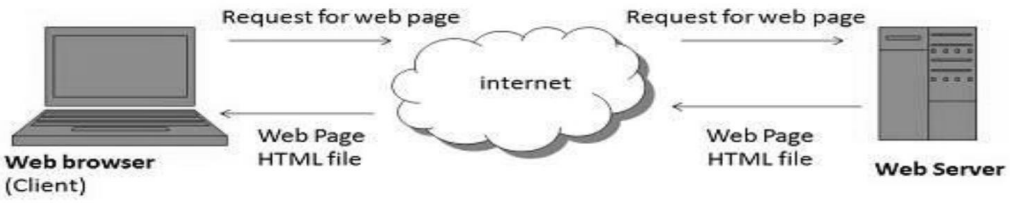


Figure : Architecture of WWW

Let us go through the scenario shown in 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 (uniform resource locator) 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.



Neethu Mathew , CSE Dept. EKCTC