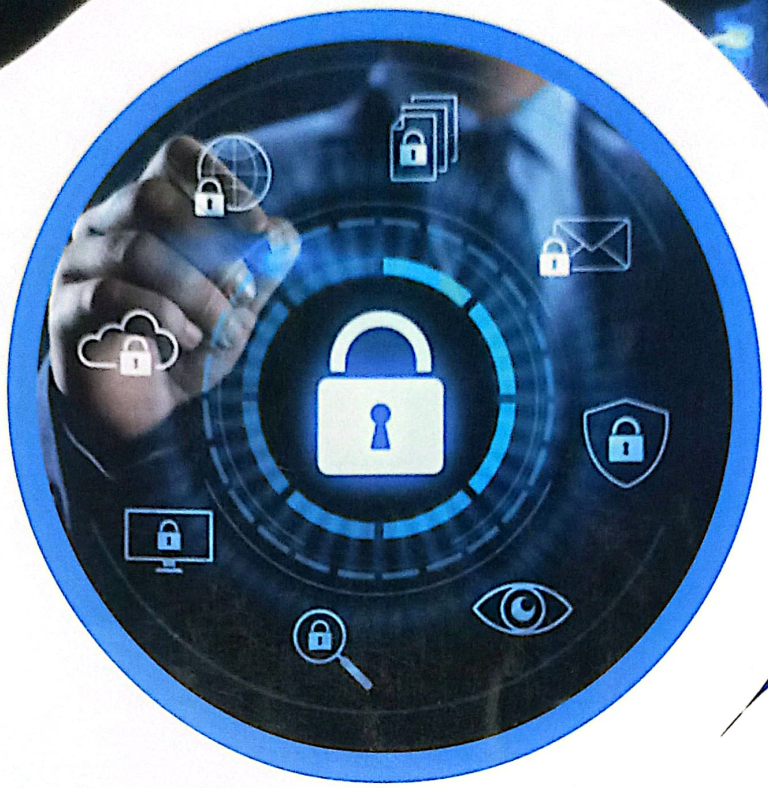
SPPU **Sem -6** Information Technology



Compulsory Subject [314451]

COMPUTER NETWORK & SECURITY

Leena R. Mehta
Cusrow Wadia Institute of Technology
(C.W.I.T.) Pune

Dr. Nilesh M. Patil
D. J. Sanghvi COE, Mumbai

- With Solved Latest UNIVERSITY QUESTION PAPERS.
- Simple and Easy Language

TECH-NEO PUBLICATIONS
Where Authors Inspire Innovation
A Sachin Shah Venture
www.techneobooks.in
info@techneobooks.in

Savitribai Phule Pune University

Computer Network & Security

(Course Code : 314451)

Semester VI : Information Technology

Strictly as per the New Syllabus (2019 Course) of Savitribai Phule Pune University w.e.f. academic year 2021-2022

Mrs. Leena B. Mahite

Dr. Nilesh M. Patil

TECH-NEO PUBLICATIONS

A SACHIN SHAH VENTURE

Computer Network & Security

(Course Code : 314451)

(SPPU - Semester VI - Information Technology

– For New Syll. 2021-2022)

Authors : Mrs. Leena R. Mehta, Dr. Nilesh M. Patil

First Edition for New Syllabus : March 2022

Tech-Neo ID : P6-55

ISBN : 978-93-5583-103-3

Published by

- **Mr. Sachin S. Shah**
  Managing Director, B. E (Industrial Electronics)
  An Alumunus of IIM Ahmedabad
- **Mrs. Nayana S. Shah, & Mr. Rahul S. Shah**

Permanent Address

Tech-Neo Publications LLP
Sr. No. 38/1, Behind Pari Company, Khedekar Industrial Estate, Narhe, Maharashtra,
Pune-411041.

Email : info@techneobooks.in
Website : www.techneobooks.in

Printed at : Image Offset (Mr. Rahul Shah)

Dugane Ind. Area, Survey No. 28/25, Dhayari Near Pari Company, Pune - 411041. Maharashtra State, India.

E-mail : rahulshahimage@gmail.com

About Managing Director... - Mr. Sachin Shah

- **Over 25 years of experience in Academic Publishing...**

  With over two and a half decades of experience in bringing out more than 1200 titles in Engineering, Polytechnic, Pharmacy, Computer Sciences and Information Technology.

- **A driven Educationalist...**

  1. B.E.(Industrial Electronics) (1992 Batch) from Bharati Vidyapeeth's College of Engineering, affiliated to University of Pune.
  2. An Alumnus of IIM Ahmedabad.
  3. A Co-Author of bestselling book on "Basic Electrical Engineering" Basic Electronics Engineering" for Degree Course in Engineering
  4. For over a decade, been working as a Consultant for Higher Education in USA and several other countries.

- **With path-breaking career...**

  - A publishing career that started with handwritten cyclostyled notes back in 1992.
  - Has to his credit, setting up and expansion of one of the leading companies in higher education publishing.

- **An experienced professional and an expert...**

  - An energetic, creative & resourceful professional with extensive experience of closely working with the best & the most eminent authors of Publishing Industry, ensures high standards of quality in contents.
  - Helping students to attain better understanding and in-depth knowledge of the subject.
  - Simplifying the methods of learning and bridging the gap between the best authors in the publishing industry and the student community for decades.

Dedicated to .....

The Readers of this Book

Authors

Preface

Dear students,

We are extremely happy to present the book of **"Computer Network & Security"** for you. We have divided the subject into small chapters so that the topics can be arranged and understood properly. The topics within the chapters have been arranged in a proper sequence to ensure smooth flow of the subject.

We are thankful to Shri. Sachin Shah for the encouragement and support that they have extended to me. We are also thankful to the staff members of Tech-Neo Publications and others for their efforts to make this book as good as it is. We have jointly made every possible efforts to eliminate all the errors in this book. However if you find any, please let us know, because that will help us to improve further.

We are also thankful to our family members and friends for their patience and encouragement.

- Authors

❑❑❑

Syllabus...

Savitribai Phule Pune University

Third Year of Information Technology (2019 Course)

314451 : Computer Network and Security

Teaching Scheme:	Credit Scheme	Examination Scheme:
Theory(TH) : 03 hrs. / week	03	Mid_Semester : 30 Marks End_Semester : 70 Marks

Prerequisite Courses, if any:

Basics of Computer Network.

Companion Course: Cyber Security

Course Objectives:

To familiarize students with-

1. The application layer services, responsibilities and protocol.
2. Fathom wireless network and different wireless standards
3. Differences in different wireless networks and to learn different mechanism used at layers of wireless network.
4. The concept of network security.
5. Basic cryptographic techniques in application development.
6. Cyber security vulnerabilities & study typical threats to modern digital systems.

Course Outcomes: On completion of the course, students will be able to–

CO1: Explain Responsibilities, services offered and protocol used at application layer of network

CO2: Apply concepts of wireless network and different wireless standards.

CO3: Recognize the Adhoc Network's MAC layer, routing protocol and Sensor network architecture. CO4: Implement the principal concepts of network security and Understand network security threats, security services, and countermeasures

CO5: Apply basic cryptographic techniques in application development.

CO6: Gain a good comprehension of the landscape of cyber security Vulnerabilities & describe typical threats to modern digital systems..

Course Contents

Unit I : APPLICATION LAYER (06 Hrs.)

Client Server Paradigm: Communication using TCP and UDP, Peer to Peer Paradigm, Application Layer Protocols: DNS, FTP, TFTP, HTTP, SMTP, POP, IMAP, MIME, DHCP, TELNET.. **(Refer chapter 1)**

Unit II : WIRELESS STANDARDS (06 Hrs.)

Wireless LANs: Fundamentals of WLAN, Design goals, Characteristics, Network Architecture, IEEE 802.11 components in IEEE 802.11 network, Physical Layer, **MAC Sub Layers :** DCF, PCF, Hidden and expose station problem, Frame format, Addressing Mechanism, IEEE 802.15.1 **Bluetooth:** Architecture Layers operational states, **IEEE 802.16 WiMax:** Services, Architecture, Layers, comparison between Bluetooth IEEE 802.11 and IEEE 802.16. **(Refer chapter 2)**

Unit III : ADHOC AND WSN (06 Hrs.)

Infrastructure Network and Infrastructure-less Wireless Networks, Issues in Adhoc Wireless Network, Adhoc Network MAC Layer: Design Issues, Design Goal, Classification, MACAW, **Adhoc Network Routing Layer:** Issues in Designing a Routing Protocol for Ad-hoc Wireless Networks – Classifications of Routing Protocols, DSDV, AODV, DSR,

Applications of Sensor Network, Comparison with Ad Hoc Wireless Network, Sensor node architecture Issues and Challenges in Designing a Sensor Network, Classification of sensor network protocols, SENSOR NETWORK ARCHITECTURE: Layered Architecture, Clustered Architecture **(Refer chapter 3)**

Unit IV : INTRODUCTION TO NETWORK SECURITY (06 Hrs.)

Importance and Need for Security, Network Attacks- Passive, Active Network Security Threats: Unauthorized access, Distributed Denial of Service (DDoS) attacks, Man in the middle attacks, **Concept of Security Principles:** Confidentiality and Privacy, Authentication, Authorization and Access Control, Integrity, Non- repudiation, Stream Ciphers: Substitution Cipher – Mono alphabetic Cipher, Polyalphabetic Substitution Cipher., Transposition Cipher: Rail-Fence

Block Ciphers modes: Electronic Code Book (ECB) Mode., Cipher Block Chaining (CBC) Mode., Cipher Feedback Mode (CFB) , Output Feedback (OFB) Mode. **(Refer chapter 4)**

Unit V : CRYPTOGRAPHIC ALGORITHM (06 Hrs.)

Mathematical preliminaries: Groups, Rings, Fields, Prime numbers, Symmetric key algorithms: Data Encryption Standards, Advanced Encryption Standard, **Public Key Encryption and Hash function:** RSA Digital signatures, Digital Certificates and Public Key Infrastructure: Private Key Management, Diffie Hellman key exchange, The PKIX Model.

(Refer chapter 5)

Unit VI : INTRODUCTION TO CYBER SECURITY (06 Hrs)

Introduction to Cyber Security: Basic Cyber Security Concepts, Layers of security, Vulnerability, Threat, Harmful Acts-Malware, Phishing, MIM Attack, DOS Attack, SQL Injection, Internet Governance – Challenges and Constraints, Computer Criminals, Assets and Threat, Motive of Attackers, Software attacks, hardware attacks, Cyber Threats-Cyber Warfare, Cyber Crime, Cyber Stalking, Cyber Terrorism, Cyber Espionage, Comprehensive Cyber Security Policy.

(Refer chapter 6)

❑❑❑

Suggested List of Laboratory Experiments/Assignments

	Group A : Computer Network
1.	Using a Network Simulator (e.g. packet tracer) Configure Router for… a) Configure a router using router commands and Configure Routing Information Protocol(RIP). b) Configure Access Control lists – Standard & Extended. c) Network Address Translation: Static, Dynamic & PAT (Port Address Translation)
2.	Using a Network Simulator (e.g. packet tracer) Configure Routing Protocols, a) Configure EIGRP – Explore Neighbor-ship Requirements and Conditions, its K Values Metrics Assignment and Calculation. b) OSPF – Explore Neighbor-ship Condition and Requirement, Neighbor-ship states, OSPF MetricCost Calculation. c) WLAN with static IP addressing and DHCP with MAC security and filters
3.	Socket Programming in C/C++ on Linux. a) TCP Client, TCP Server b) UDP Client, UDP Server
4	Introduction to server administration (server administration commands and their applications) and configuration of below Server: (Study/Demonstration Only) a) FTP b) Web Server
	Group B : Network Security
1.	Implement a client and a server on different computers using python. Perform the communication between these two entities by using RSA cryptosystem
2.	Implement a client and a server on different computers using python. Perform the authentication of sender between these two entities by using RSA digital signature cryptosystem.
3.	Implement a client and a server on different computers using python. Perform the encryption of message of sender between these two entities by using DES Algorithm and use Diffie Hellman method for exchange of keys.
4.	Use the snort intrusion detection package to analyze traffic and create a signature to identify problem traffic.

❑❑❑

Table of Contents

In Sem

End Sem

Unit I

CHAPTER 1

Application Layer

Syllabus

Client Server Paradigm: Communication using TCP and UDP, Peer to Peer Paradigm, Application Layer Protocols: DNS, FTP, TFTP, HTTP, SMTP, POP, IMAP, MIME, DHCP, TELNET.

1.1 INTRODUCTION TO APPLICATION LAYER

- Application layer is **the topmost layer** of OSI Model. This layer provides several services for an application program to enable it to **establish connection** with another application program for communication purpose in the environment of a network.
- Practically the application layer is not an application; rather it is a **component within an application**, which handles the respective communication method used to communicate with other devices.
- Application layer can be considered as an **abstraction layer service**, which helps to mask the remaining applications from the transmission process.
- The application layer in a practical manner **depends upon all the layers**, which are present below it to complete its process. Here, **visual interface** of the data or the application is used, so that the user can understand it easily. And can verify that there is a presence of required communication interfaces.
- **For example**: checking the presence of an Ethernet or Wi-Fi interface in the sender's computer.
- Verify agreement at both the connection ends regarding **error recovery procedures**, **data integrity** as well as **privacy**. Sets protocol and rules regarding **data syntax** at the application level. Presents the data on the receiving end of the user application.

☞ **Application layer protocols**

- File Transfer Protocol (FTP),
- Trivial File Transfer Protocol (TFTP),
- Simple Mail Transfer Protocol (SMTP),
- Telnet,
- Domain Name System (DNS) etc.

1.2 RESPONSIBILITIES OF APPLICATION LAYER

UQ. What are the main responsibilities of Application Layer? Explain in brief. **(SPPU- Q. 1(b), August 14, 04 Marks)**

1. Network Virtual Terminal
2. File Transfer, Access and Management
3. Addressing
4. Mail Services
5. Directory Services
6. Authentication

1. Network Virtual Terminal

- The application layer is the **software version of a physical terminal** and this layer permits a user to log on to a remote host.
- For this, an application creates a **software emulation of a terminal at the remote host**. By this user's computer can communicate with the software terminal, which in turn, communicates with the host. Means the remote host is communicating with one of its terminals, so it allows the user to log on.

2. File Transfer, Access, and Management (FTAM)

- An application permits a user to access files in a remote computer, to retrieve files from a computer and to manage files on a remote computer.
- FTAM is concerned with a hierarchical virtual file in terms of file attributes, file structure and the types of operations performed on the files and their attributes.

3. Addressing

- To achieve communication between client and server system, there is a need for addressing. When a request is sent from the client side to the server side, this request contains the server address and its own address.
- The server answered to the client request, this request contains the destination address, **i.e.,** client address. **DNS** is used to achieve this type of addressing.

4. Mail Services

Email forwarding and storage of **e-mails** provided by an application layer.

5. Directory Services

A **distributed database** is contained by an application that provides access for global information about various objects and services.

6. Authentication

It provides authentication to occur between devices for an **extra layer of security** and it authenticates the sender or receiver's message or both.

1.3 TYPES OF COMMUNICATION PARADIGMS

The remote application processes can communicate mainly in two different fashions :

- **Peer-to-peer :** Both remote processes are executing at same level and they exchange data using some shared resource.
- **Client-Server :** One remote process acts as a Client and requests some resource from another application process acting as Server.

1.4 CLIENT SERVER PARADIGM

- The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients.
- Server applications are capable of providing services to large number of clients e.g. Google's Gmail server is capable of serving large number of Gmail clients (web client or mobile client etc).
- Client and server may be residing at different geographical locations and connected to each other via internet like Facebook server may be running (or hosted) in USA and we can be able to access Facebook using web client from Singapore.
- In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client. Clients do not share any of their resources.
- Examples of Client-Server Model are Email, World Wide Web, etc.

1.4.1 Working of Client-Server Model

GQ. Explain working of Client-Server Model.

One of the **Client-Server** models is the **Internet** access via, web browsers. Shown in Fig. 1.4.1.

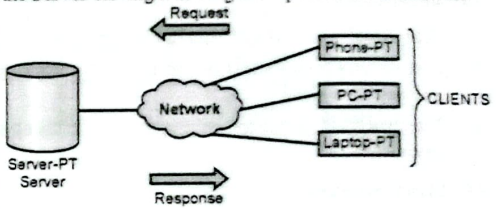
☞ **Client**

- When we talk the word **Client**, it means to talk of a person or an organization using a particular service.
- Similarly in the digital world a **client** is a computer (**Host**) i.e. capable of receiving information or using a particular service from the service providers (**Servers**).

☞ **Servers**

- Similarly, when we talk the word **Servers**, it mean a person or medium that serves something. Shown in Fig. 1.4.1.
- Similarly in this digital world a **Server** is a remote computer which provides information (data) or access to particular services.

So, it's basically the **Client** requesting something and the **Server** serving it as long as its present in the database.

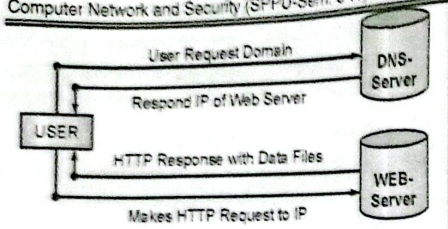


(IA1)**Fig. 1.4.1 : Client-Server**

1.4.2 Interaction of Browser with the Servers

There are few steps to follow to interacts with the servers a client.

- User enters the **URL** (Uniform Resource Locator) of the website or file. The Browser then requests the **DNS** (DOMAIN NAME SYSTEM) Server. Shown in Fig. 1.4.2.
- **DNS Server** lookup for the address of the **WEB Server.**
- DNS Server responds with the IP address of the WEB Server.
- Browser sends over an **HTTP/HTTPS** request to **WEB Server's IP** (provided by **DNS server**).
- Server sends over the necessary files of the website.
- Browser then renders the files, and the website is displayed. This rendering is done with the help of **DOM** (Document Object Model) interpreter, **CSS** interpreter and **JS Engine** collectively known as the **JIT** or (Just in Time) Compilers.



(1A2) Fig. 1.4.2 : Browser and Server Interaction

1.4.3 Advantages and Disadvantage of Client-Server Model

Advantages

1. Centralized system with all data in a single place.
2. Cost efficient requires less maintenance cost and Data recovery is possible.
3. The capacity of the Client and Servers can be changed separately.

Disadvantages

1. Clients are prone to viruses, Trojans and worms if present in the Server or uploaded into the Server.
2. Server are prone to Denial of Service (DOS) attacks.
3. Data packets may be spoofed or modified during transmission.
4. Phishing or capturing login credentials or other useful information of the user are common and MITM (Man in the Middle) attacks are common.

1.5 CONCURRENCY

GQ. Write short note on Concurrency.

UQ. List and Explain different types of Servers.

(SPPU- Q. 3(b), August 14, 04 Marks)

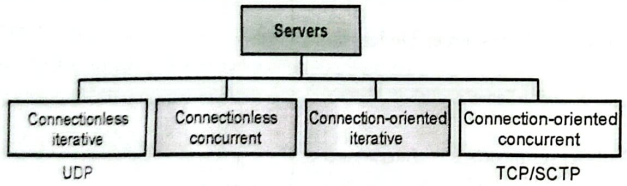
Both clients and servers can run in concurrent mode.

(a) Concurrency in Clients

- Clients can be run on a machine either iteratively or concurrently.
- *Iteratively running of client* means running them one by one; one client must start, run, and terminate before the machine can start another client.
- Most computers today, however, allow *concurrent* clients; that is, two or more clients can run at the same time.

(b) Concurrency in Servers

- An *iterative* server can process only one request at a time;
  - it receives a request,
  - processes it, and
  - sends the response to the requestor before it handles another request.
- A concurrent server, on the other hand, can process many requests at the same time and thus can share its time between many requests.
- The servers use either a connectionless transport layer protocol like UDP, or a connection-oriented transport layer protocol like TCP/ SCTP.
- Server operation, therefore,depends on two factors :
  - the transport layer protocol and
  - the service method.
- There are four types of servers :
  - connectionless iterative,
  - connectionless concurrent,
  - connection-oriented iterative, and
  - connection oriented concurrent
  - shown in Fig. 1.5.1.

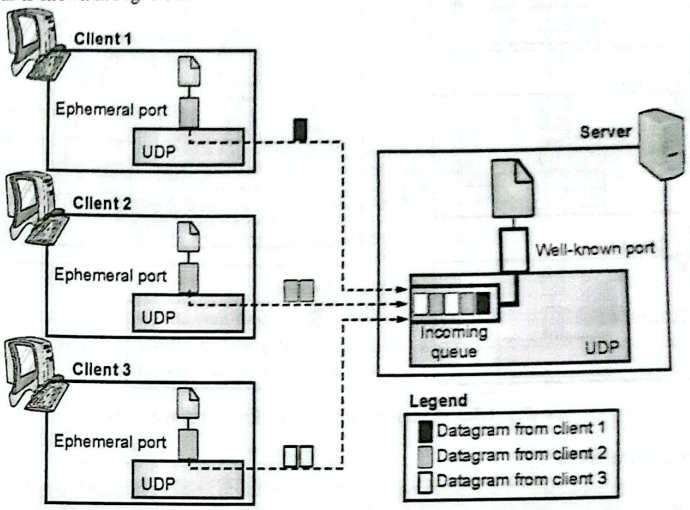


(1A3) Fig. 1.5.1 : Types of Servers

(c) Connectionless Iterative Server

- The iterative servers normally use UDP protocol, this server processes one request at a time.
- A server gets the request received in a datagram from UDP, processes the request, and gives the response to UDP to send to the client.

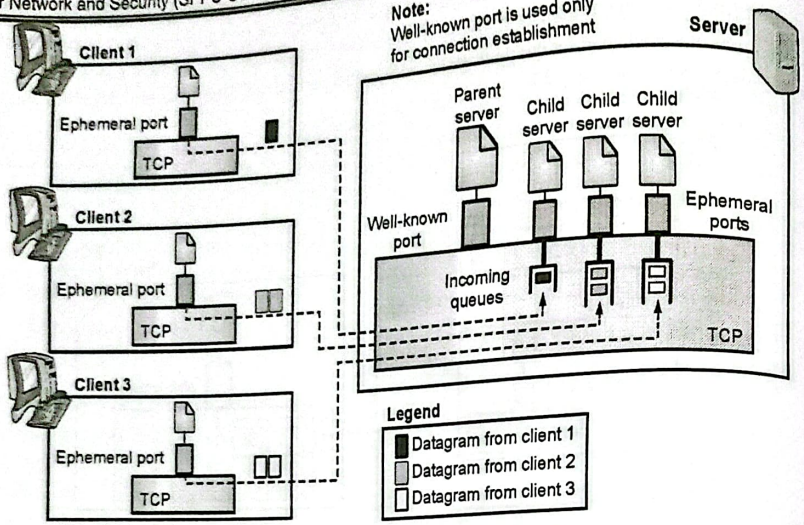
- The server pays no attention to the other datagrams. These datagrams are stored in a queue, waiting for service. They could all be from one client or from many clients. In either case they are processed one by one in order of arrival.
- The server uses one single port for this purpose, the well-known port. All the data-grams arriving at this port wait in line to be served, as is shown in Fig. 1.5.2.



(1A4) Fig. 1.5.2 : Connection Iterative Server

(d) Connection-Oriented Concurrent Server

- The concurrent servers generally use TCP (or SCTP). The server can serve many clients at the same time.
- The communication is connection-oriented, which means that a request is a stream of bytes that can arrive in several segments and the response can occupy several segments.
- A connection is established between the server and each client, and the connection remains open until the entire stream is processed and the connection is terminated
- In this type of server cannot use only one port because each connection needs a port,and many connections may be open at the same time. Many ports are needed, but a servercan use only one well-known port.
- The solution is to have one well-known port and many ephemeral ports. The server accepts connection requests at the well-known port. A client cnmake its initial approach to this port to make the connection. After the connection is made, the server assigns a temporary port to this connection to free the well-known port.
- Data transfer can now take place between these two temporary ports, one at the client site and the other at the server site. The well-known port is now free for another client to makethe connection.
- To serve several clients at the same time, a server creates child processes,which are copies of the original process (parent process).
- The server must also have one queue for each connection. The segments come from the client, are stored in the appropriate queue, and will be served concurrently bythe server. Shown in Fig. 1.5.3 for this configuration.

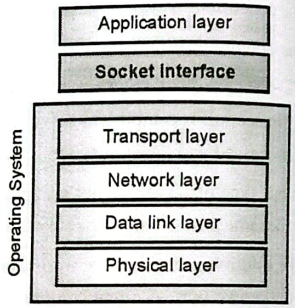


(1A5)**Fig. 1.5.3 : Connection Oriented Concurrent Server**

1.5.1 Socket Interfaces

UQ. Explain the terms ;
1. Socket 2. Data Structure 3. Structure of Socket Address **(SPPU- Q. 3(a), August 15, 06 Marks)**

- A computer program is a set of predefined instructions that tells the computer what to do. A computer program has a set of instructions for mathematical operations, another set of instructions for string manipulation, still another set of instructions for input/output access.
- If we need a program to be able to communicate with another program running on another machine, we need a new set of instructions to tell the transport layer to open the connection, send data to and receive data from the other end, and close the connection.
- A set of instructions of this kind is normally referred to as an **interface.** Several interfaces have been designed for communication.
- The Three among them are common:
  - Socket interface,
  - transport layer interface (TLI), and
  - STREAM.



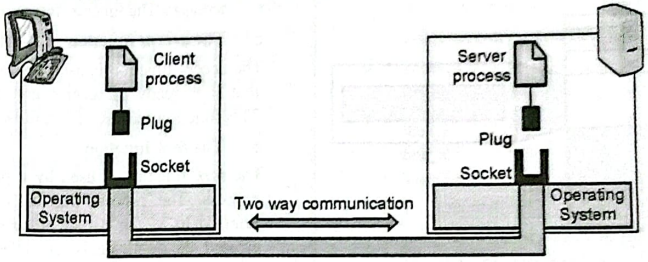
(1A6)**Fig. 1.5.4 : Relation Between the Operating System and the TCP/IP Suite**

- The socket interface, as a set of instructions, is located between the operating system and the application programs.
- To access the services provided by the TCP/IP protocol suite, an application needs to use the instructions defined in the socketinterface.

(a) Socket

- A **socket** is a software abstract simulating a hardware socket we see in our daily life. To use the communication channel, an application program (client or server) needs to request the operating system to create a socket.
- The application program then can *plug* into the socket to send and receive data. For data communication to occur, a pair of sockets, each at one end of communication, is needed.

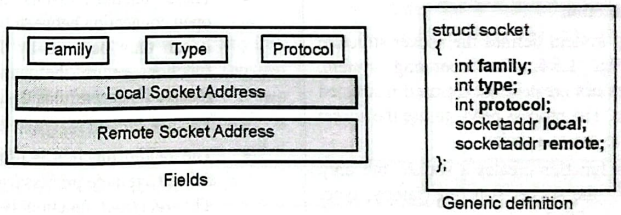
- Fig. 1.5.5 simulates this abstraction using the socket and plug that we use in our daily life (for a telephone, for example); on the Internet a socket is a software data structure.



(1A7)**Fig. 1.5.5 : Concept of Sockets**

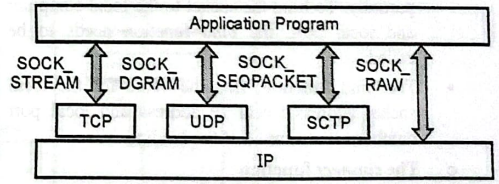
(b) Data Structure

- The format of data structure to define a socket depends on the underlying language used by the processes.
- The processes are written in C language. In C language, a socket is defined as a five-field structure (structor record) as shown in Fig. 1.5.6.



(1A8)**Fig. 1.5.6 : Socket Data Structure**

- The programmer should not redefine this structure; it is already defined. The programmer needs only to use the header file that includes this definition. The field uses in this structure are:
  - **Family.** This field defines the protocol group: IPv4, IPv6, UNIX domain protocols, and so on. The family type we use in TCP/IP is defined by the constant IF_INET for IPv4 protocol and IF_INET6 for IPv6 protocol.
  - **Type.** This field defines four types of sockets: SOCK_STREAM (for TCP), SOCK_DGRAM (for UDP), SOCK_SEQPACKET (for SCTP), and SOCK_RAW (for applications that directly use the services of IP. They are shown in Fig. 1.5.7.
  - **Protocol.** This field defines the protocol that uses the interface. It is set to 0 for TCP/IP protocol suite.
  - **Local socket address.** This field defines the local socket address. A socket address is a combination of an IP address and a port number.
  - **Remote socket address.** This field defines the remote socket address.

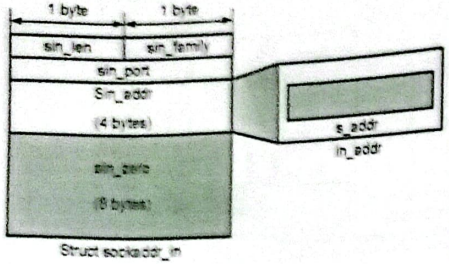


(1A9)**Fig. 1.5.7 : Socket Types**

(c) Structure of a Socket Address

The structure of a socket address is a combination of IP address and port number. Although several types of socket addresses have been defined in network programming, the one used for IPv4 is discussed. In

this version a socket address is a complex data structure (struct in C) as shown in Fig. 1.5.8.



(1A10) **Fig. 1.5.8 : IPV4 socket**

(d) Functions

The interaction between a process and the operating system is done through a list of predefined functions. Functions are combined to create processes. Different functions are as follows :

- **The *socket* function**
- The operating system defines the socket structure shown in Fig. 1.5.6. The operating system, however, does not create a socket until instructed by the process. The process needs to use the *socket* function call to create a socket.
- A call to this function creates a socket, but only three fields in the socket structure (family, type, and protocol) are filled.
- If the call is successful, the function returns a unique socket descriptor *sockfd* (a non-negative integer) that can be used to refer to the socket in other calls; if the call is not successful, the operating system returns –1.
- **The *bind* function**
- The socket function fills the fields in the socket partially. To bind the socket to the local computer and local port, the *bind* function needs to be called.
- The bind function, fills the value for the local socket address (local IP address and local port number). It returns –1 if the binding fails.
- **The *connect* function**
- The connect function is used to add the remote socket address to the socket structure. It returns –1 if the connection fails.
- **The *listen* function**
- The listen function is called only by the TCP server. After TCP has created and bound a socket, it must inform the operating system that a socket is ready for receiving clientrequests.
- This is done by calling the *listen* function. The backlog is the maximum number of connection requests. The function returns –1 if it fails.
- **The *accept* function**

The accept function is used by a server to inform TCP that it is ready to receive connections from clients. This function returns –1 if it fails.

- **The *fork* function**

The *fork* function is used by a process to duplicate a process. The process that calls the *fork* function is referred to as the parent process; the process that is created, the duplicate, is called the child process.

- **The *send* and *recv* functions**
- The *send* function is used by a process to send data to another process running on a remote machine. The *recv* function is used by a process to receive data from another process running on a remote machine.
- These functions assume that there is already an open connection between two machines; therefore, it can only be used by TCP (or SCTP). These functions return the number of bytes send or receive if successful and –1 if there is an error.
- **The *sendto* and *recvfrom* functions**
- The *sendto* function is used by a process to send data to a remote process using theservices of UDP. The *recvfrom* function is used by a process to receive data from a remote process using the services of UDP.
- Since UDP is a connectionless protocol, one of the arguments defines the remote socket address (destination or source). These functions return the number of bytes sent or received if successful and –1 if there is an error.
- **The *close* function**

The close function is used by a process to close a socket. The socket returns an integer, 0 for success and –1 for error.

- **Byte ordering functions**
- Information in a computer is stored in *host byte order*, which can be *little-endian*, in which the little-end byte (least significant byte) is stored in the starting address, or *big-endian*, in which the big-end byte (most significant byte) is stored in the starting address.
- Network programming needs data and other pieces of information to be in *network byte order*, which is big-endian. Since when we write programs, we are not sure, how the information such as IP addresses and port number are stored in the

computer, we need to change them to network byte order.

- Two functions are designed for this purpose: ***htons*** (host to network short), which changes ashort (16-bit) value to a network byte order, and ***htonl*** (host to network long), which does the same for a long (32-bit) value. There are also two functions that do exactly the opposite: ***ntohs*** and ***ntohl.***
- **Memory management functions**
- Some functions are needed to manage values stored in the memory. Three common memory used functions are.
- The first function, *memset* (memory set) is used to set (store) a specified number of bytes (value of *len*) in the memory defined by the destination pointer (starting address).
- The second function, *memcpy* (memory copy) is used to copy a specified number of bytes (value of *nbytes*) from part of a memory (source) to another part of memory (destination).
- The third function, *memcmp* (memory compare), is used to compare two sets of bytes (nbytes) starting from ptr1 and ptr2. The result is 0 if two sets are equal, less than zero if the first set is smaller than the second, and greater than zero if the first set is larger than the second. The comparison is based on comparing strings of bytes in the C language.
- **Address conversion functions**
- Normally 32-bit IP address in dotted decimal format is used. When we want to store the address in a socket, however, we need to change it to a number.
- Two functions are used to convert an address from a presentation to a number and vice versa: ***inet_pton*** (presentation to number) and ***inet_ntop*** (number to presentation). The constant use for family value is AF_INET for our purpose.

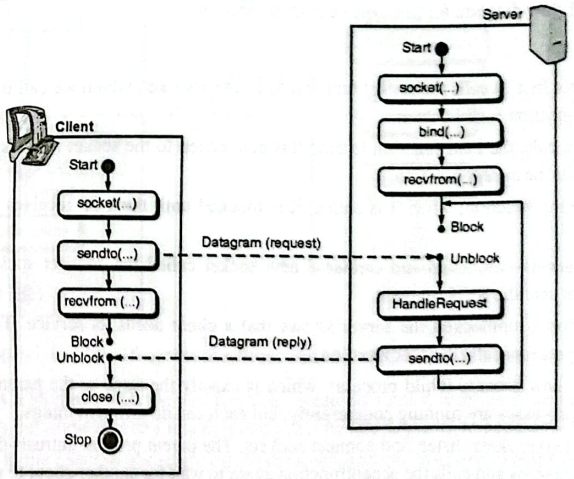
(e) Header Files

- A header file is needed so the this described functions can be used. A separate file, "headerFiles.h" is defined.
- This file is included in our programs to avoid including long lists of header files.
- Not all of these header files maybe needed in all programs, but it is recommended to include all of them in case.

1.6 COMMUNICATION USING UDP

GQ. Describe how communication take place using UDP.

Fig. 1.6.1 shows a simplified flow diagram for this type of communication.



(1A11) **Fig. 1.6.1 : Connectionless Iterative Communication using UDP**

Server Process

- The server process starts first. The server process calls the *socket* function to create a socket.
- It then calls the *bind* function to bind the socket to its well-known port and the IP address of the computer on which the server process is running.
- The server then calls the *recvfrom* function, which blocks until a datagram arrives.
- When the datagram arrives, the *recvfrom* function unblocks, extracts the client socket address and address length from the received datagram, and returns them to the process.
- The process saves these two pieces of information and calls a procedure (function) to handle the request. When the result is ready, the server process calls the *sendto* function and uses the savedinformation to send the result to the client that requested it.
- The server uses an infinite loop to respond to the requests coming from the same client or different clients.

Client Process

- The client process is simpler. The client calls the *socket* function to create a socket.
- It then calls the *sendto* function and pass the socket address of the server and the locationof the buffer from which UDP can get the data to make the datagram.
- The client then calls a *recvfrom* function call that blocks until the reply arrives from the server. When thereply arrives, UDP delivers the data to the client process, which make the recv functionto unblock and deliver the data received to the client process.
- Note that we assume thatthe client message is so small that it fits into one single datagram. If this is not the case, the two function calls, *sendto* and *recvfrom*, need to be repeated.
- However, the server isnot aware of multi datagram communication; it handles each request separately.

1.7 COMMUNICATION USING TCP

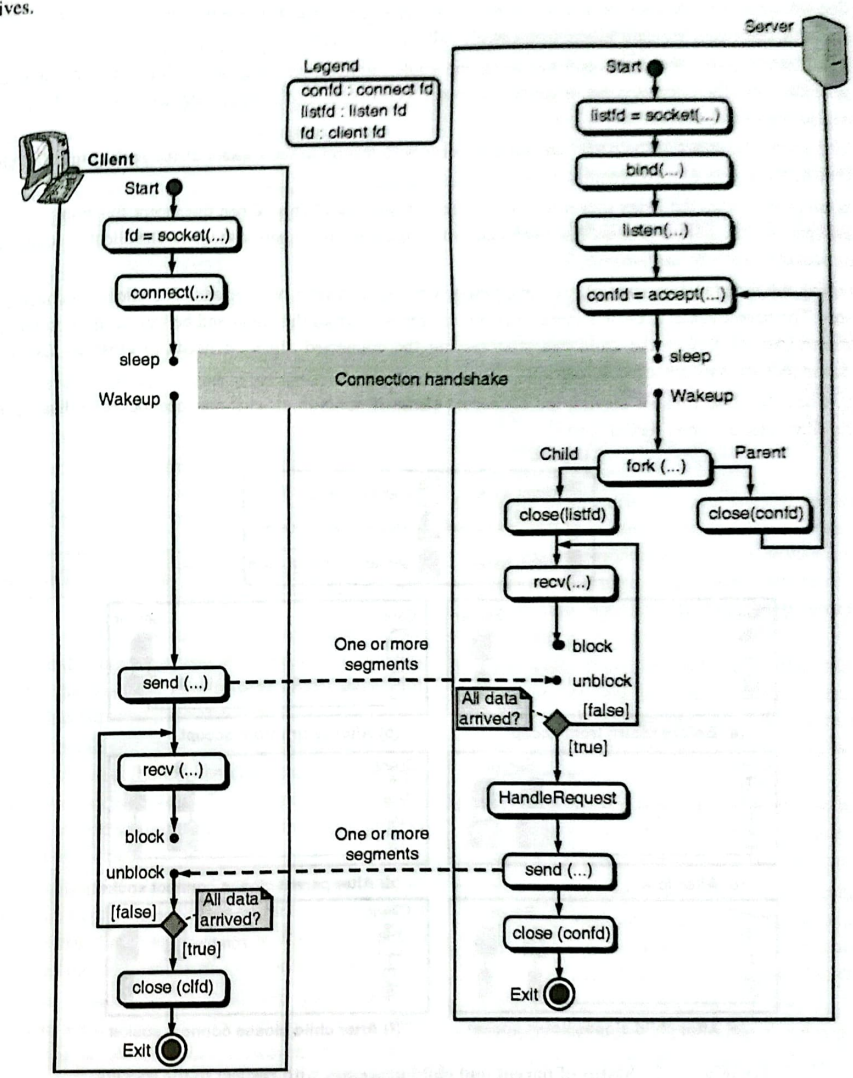
GQ. Describe how communication take place using TCP.

Connection-oriented, concurrent communication using the service of TCP (the case of SCTP would be similar). Fig. 1.7.1 shows the general flow diagram for this type of communication.

Server Process

- The server process starts first. It calls the *socket* function to create a socket, which we call the *listen* socket. This socket is only used during connection establishment.
- The server process then calls the *bind* function to bind this connection to the socket addressof the server computer. The server program then calls the *accept* function.
- This functionis a blocking function; when it is called, it is blocked until the TCP receives a connection request (SYN segment) from a client.
- The *accept* function then is unblocked and creates a new socket called the connect socket that includes the socket address of the client that sent the SYN segment.
- After the *accept* function is unblocked, the server knows that a client needs its service. To provide concurrency, the server process (parent process) calls the *fork* function.
- This function creates a new process (child process), which is exactly the same as the parent process. After calling the *fork* function, the two processes are running concurrently, but each can do different things.
- Each process now has two sockets: listen and connect sockets. The parent process entrusts the duty of serving the client to the hand of the child process and calls the acceptfunction again to wait for another client to request connection.

- The child process is nowready to serve the client. It first closes the listen socket and calls the *recv* function to receive data from the client. The *recv* function, like the *recvfrom* function, is a blocking function; it is blocked until a segment arrives.

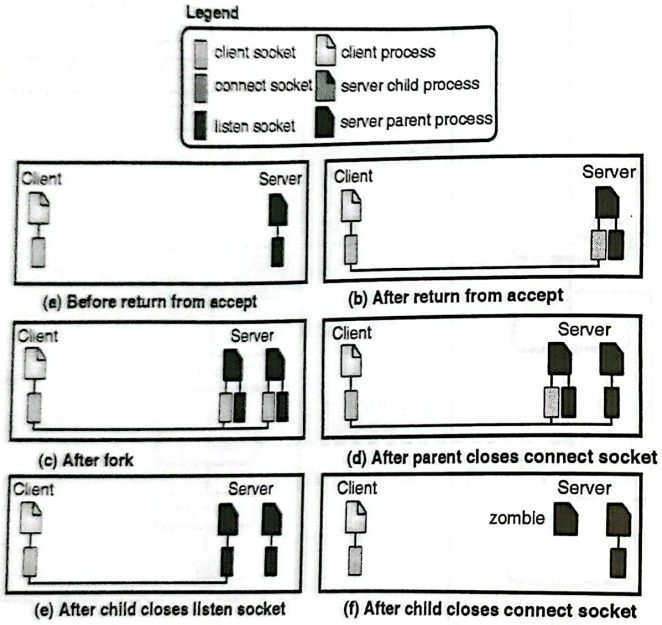


(1A12)**Fig. 1.7.1 : Flow Diagram of connection oriented Concurrent communication**

- The child process uses a loop and calls the *recv* function repeatedly until it receives all segments sent by the client. The child processthen gives the whole data to a function (we call it *handleRequest*), to handle the requestand return the result.
- The result is then sent to the client in one single call to the *send* function. We need to emphasize several points here.

- First, the flow diagram we are using is the simplest possible one. The server may use many other functions to receive and senddata, choosing the one which is appropriate for a particular application.
- Second, we assume that size of data to be sent to the client is so small that can be sent in one singlecall to the *send* function, otherwise, we need a loop to repeatedly call the *send* function.
- Third, although the server may send data using one single call to the *send* function, TCPmay use several segments to send the data. The client process, therefore, may not receive data in one single segment, as we will see when we explain the client process.

- Fig. 1.7.2 shows the status of the parent and child process with respect to the sockets. Part a in the Fig. 1.7.2 shows the status before the *accept* function returns.
- The parent process uses the *listen* socket to wait for request from the clients. When the accept function is blocked and returned (part b), the parent process has two sockets:the *listen* and the *connect* sockets. The client is connected to the *connect* socket.
- Aftercalling the *fork* function (part c), we have two processes, each with two sockets. The client is connected to both processes. The parent needs to close it *connect* socket to free itself from the client and be free to listen to requests from other clients (part d). Before the child can start serving the connected client, it needs to close its *listen* socket so that a future request does not affect it (part e).
- Finally, when the child finishes serving the connected client, it needs to close it connect socket to disassociate itself from the client that has been served (part f).



(1A13)**Fig. 1.7.2 : Status of parent and child processes with respect to the sockets**

Although we do not include the operation in our program (for simplicity), the child process, after serving the corresponding process, needs to be destroyed.

- The child process that has done its duty and is dormant is normally called a zombie in the UNIX environment system.
- A child can be destroyed as soon as it is not needed. Alternatively, the system can run a special program once in a while to destroy all zombies in the system.
- The zombies occupy space in the system and can affect the performance of the system.

Client Process

- The client process is simpler. The client calls the *socket* function to create a socket. It then calls the *connect* function to request a connection to the server.
- The *connect* function is a blocking function; it is blocked until the connection is established between two TCPs. When the *connect* function returns, the client calls the *send* function to send datato the server.
- We use only one call to the *send* function, assuming that data can be sent with one call. Based on the type of the application, we may need to call this function repeatedly (in a loop).
- The client then calls the *recv* function, which is blocked until a segment arrives and data are delivered to the process by TCP. Note that, although the data are sent by the server in one single call to the *send* function, the TCP at the server site may have used several segments to send data.
- This means we may need to call the *recv* function repeatedly to receive all data. The loop can be controlled by the return value of the *recv* function.

1.8 PEER-TO-PEER PARADIGM (P2P)

- Although most of the applications available on the Internet today use the client-server paradigm, the idea of using the so-called **peer-to-peer (P2P) paradigm** recently has attracted some attention.
- In this paradigm, two peer computers (laptops, desktops, or main frames) can communicate with each other to exchange services. There is no need of server process , instead the responsibility of the server process is shared by the peer.
- This paradigm is interesting in some areas such file as transfer in which the client-server paradigm may put a lot of the load on the server machine if a client wants to transfer a large file such as an audio or video file.
- The idea is also interesting if two peers need to exchange some files or information to each other without going to a server. However, we need to mention that the P2P paradigm does not ignore the client-server paradigm. What it does actually is to let the duty of a server be shared by some users that want to participate in the process.
- For example, instead of allowing several clients to make a connection and each download a large file, a server can let each client download a part of a file and then share it with each other.
- In the process of downloading part of the file or sharing the downloaded file, however, a computer needs to play the role of a client and the other therole of a server.
- In other words, a computer can be a client for a specific application at one moment and the server at another moment. These applications are now controlled commercially and not formally part of the Internet.

1.8.1 Advantages and Disadvantages of Peer-to-Peer Networks

Advantages

1. **Low latency :** With low latency come better response times and shorter waiting times between requests. The length of the connection path that is between peers is reduced, making the network more efficient by eliminating redundant steps on the way towards the final destination.
2. **High bandwidth :** A peer-to-peer network provides you with high bandwidth, so there is no need for central servers to provision resources. This makes it possible to provide information to a huge number of users at the same time without affecting the performance of the network.
3. **Low cost :** Due to the fact that there is no central server in a peer-to-peer network, each peer is responsible for storing and sending the requested information. There are no fees charged by the server that is hosting the application.
4. **High security :** A peer-to-peer network has no single point of failure. If one node goes offline or becomes unavailable, the network is still able to function.
5. **Decentralization :** In a client-server architecture, the company that owns the server controls its users. It can monitor user activity and delete information. That is not the case when it comes to peer-to-peer networks, which let users control their own data.
6. **Fault tolerance :** Peer-to-peer networks are very resilient to faults. When one node goes down, other nodes will cover for this node and keep the network running.

Disadvantages

1. **Scalability :** Scalability is an issue that affects peer-to-peer networks. It is not an issue with client-server networks, though. Since it is difficult to add new peers

to the network and maintain great performance levels at the same time. There are several ways to solve this problem, including upgrading the hardware. However, it tends to be more expensive than using a client-server architecture.

2. **Security concerns :** Peer-to-peer networks, such as file-sharing applications, are places where you can easily get infected with malware. Caution is advised.

1.8.2 Difference Between P2P and Client-Server

UQ. State difference between Client-Server and Peer-to-Peer Network **(SPPU- Q. 5(a), May 15, 10 Marks, Q. 10(c), May 15, 08 Marks)**

Parameters	CLIENT-SERVER	PEER-TO-PEER
Basic	There is a specific server and specific clients connected to the server.	Clients and server are not distinguished; each node act as client and server.
Service	The client request for service and server respond with the service.	Each node can request for services and can also provide the services.
Focus	Sharing the information.	Connectivity.
Data	The data is stored in a centralized server.	Each peer has its own data.
Server	When several clients request for the services simultaneously, a server can get bottlenecked.	As the services are provided by several servers distributed in the peer-to-peer system, a server in not bottlenecked.
Expense	The client-server are expensive to implement.	Peer-to-peer are less expensive to implement.
Stability	Client-Server is more stable and scalable.	Peer-to-Peer suffers if the number of peers increases in the system.

1.9 WEB (WWW)

GQ. Write short note on www and internet.

- The Web is the common name for the **World Wide Web**, a subset of the Internet consisting of the pages that can be accessed by a **Web browser**. The Web and the Internet terms are used interchangeably.
- However, the term Internet actually refers to the **global network of servers** that makes the information sharing that happens over the Web possible. The Web does make up a large portion of the Internet, but web and internet are not the same.
- The World Wide Web (**WWW**) is a **repository of information** linked together from points all over the world. The WWW has a unique combination of flexibility, portability, and user-friendly features that distinguish it from other services provided by the Internet.
- The WWW project was initiated by CERN (European Laboratory for Particle Physics) to create a system to handle distributed resources necessary for scientific research.

1.9.1 Architecture of WWW

GQ. Explain architecture of WWW.

- The WWW today 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** Fig. 1.9.1.
- 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 page scan be retrieved and viewed by using browsers shown in Fig. 1.9.1.

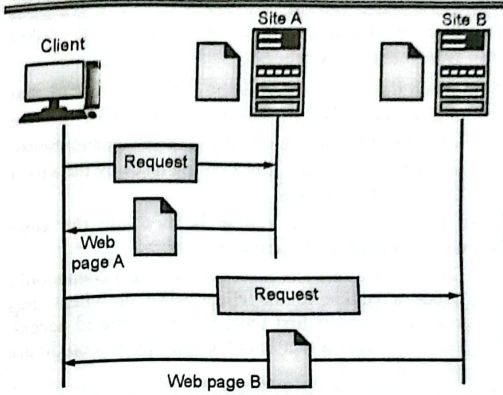


Fig. 1.9.1 : Architecture of WWW

- 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 Webpage, called the **Uniform Resource Locator (URL).**
- The server at **site A** finds the document and sends it to the client. When the user views the document, 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.

1.9.1(A) Client (Browser)

Browsers are means 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.
- The client protocol can be one of the protocols described previously such as **FTP or HTTP**. The interpreter can be HTML, Java, or JavaScript, depending on the type of document. As shown in Fig. 1.9.2.

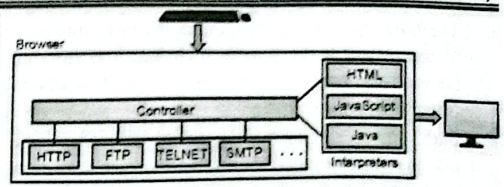


Fig. 1.9.2 : Browser

1.9.1(B) 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 memory;** this memory is faster to access than disk.
- A server can also become more efficient through **multithreading or multiprocessing**. In this case, a server can answer more than one request at a time.

1.9.1(C) Uniform Resource Locator (URL)

GQ. Explain URL

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**. As shown in Fig. 1.9.3.

Fig. 1.9.3 : URL

- The ***protocol*** is the client/server program used to retrieve the document. Many different protocols can retrieve a document; among them is **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 aliased. (Giving other name)
- **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 Webpage.

- 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 :** It is the path name 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 sub directories and files.

1.9.1(D) Cookies

GQ. Explain Cookies.
UQ. What are Cookies?
SPPU- Q. 6(b), Oct. 16, 02 Marks

The World Wide Web was originally designed as a **stateless entity**. A client sends a request; a server responds. Their relationship is over. The original design of WWW retrieving publicly available documents exactly fits this purpose.

Today the Web has other functions; some are listed here:

1. Some websites need to allow access to registered clients only.
2. Websites are being used as an electronic store that allow users to browse through the store, select wanted items, put them in an electronic cart, and pay at the end with a credit card.
3. Some websites are used as portal: the user selects the Webpages he wants to see.
4. Some websites are just used for advertising purpose.
5. When a server receives a request from a client, it stores information about the client in a file or a **string**. The information may include the domain name of the **client, the contents of the cookie** (information the server has gathered about the client such as name, registration number, and so on), **a timestamp**, and **other information** depending on the implementation.
6. The server includes the cookie in the response that it sends to the client.
7. When the client receives the response, the browser stores the cookie in the **cook i.e. directory**, which is sorted by the domain server name.

Working of Cookies

- When a client sends a request to a server, the browser looks in the cookie directory to see if it can find a cookie sent by that server. If found, the cookie is included in the request.
- When the server receives the request, it knows that this is an old client, not a new one. Note that the contents of the cookie are never read by the browser or disclosed to the user. It is a cookie made by the server and eaten by the server.

Now let us see how a cookie is used for the four previously mentioned purposes:

1. The site that restricts access to registered clients creates sends a cookie to the client when the client registered for the first time. For any repeated access only those clients that send the appropriate cookie are allowed.
2. An electronic store (**e-commerce**) can use a cookie for its client shoppers. When a client selects an item and inserts it into a cart, a cookie that contains information about the item, such as its number and unit price, is sent to the browser. If the client selects a second item, the cookie is updated with the new selection information. And so on. When the client finishes shopping and wants to check out, the last cookie is retrieved, and the total charge is calculated.
3. A Web portal uses the cookie in a similar way. When a user selects her favourite pages, a cookie is made and sent. If the site is accessed again, the cookie is sent to the server to show what the client is looking for.
4. A cookie is also used by advertising agencies. An advertising agency can place banner ads on some main website that is often visited by users. The advertising agency supplies only a **URL** that gives the banner address instead of the banner itself. When a user visits the main website and clicks on the icon of an advertised corporation, a request is sent to the advertising agency. The advertising agency sends the banner, **a GIF file**, for example, but it also includes a cookie with the ill of the user.

1.9.1(E) Webpages

UQ. Explain static and Dynamic Web pages.
SPPU- Q. 4(b), Dec. 16, 04 Marks

- Web page is a document available on world wide web. Web Pages are stored on web server and can be viewed using a web browser.
- A web page can contain huge information including text, graphics, audio, video and hyper links. These hyper links are the link to other web pages.

- Collection of linked web pages on a web server is known as website. There is unique Uniform Resource Locator (URL) is associated with each web page.

Static Web page

- Static web pages are also known as flat or stationary web page. They are loaded on the client's browser as exactly they are stored on the web server.
- Such web pages contain only static information. User can only read the information but can't do any modification or interact with the information.
- Static web pages are created using only HTML. Static web pages are only used when the information is no more required to be modified.

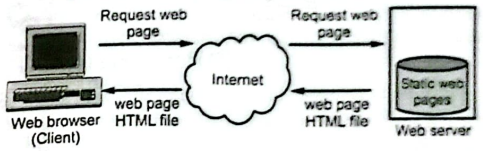


Fig. 1.9.4 : Static Web Page

Dynamic Web page

- Dynamic web page shows different information at different point of time.
- It is possible to change a portion of a web page without loading the entire web page. It has been made possible using Ajax technology.

Server-side dynamic web page

- It is created by using server-side scripting. There are server-side scripting parameters that determine how to assemble a new web page which also include setting up of more client-side processing.

Client-side dynamic web page

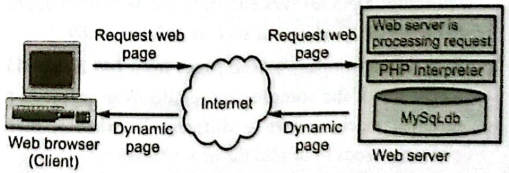


Fig. 1.9.5 : Dynamic Webpage

It is processed using client side scripting such as JavaScript. And then passed in to Document Object Model (DOM).

1.10 WEB CACHING

GQ. Write short note on Web Caching.

- Web caching is the activity of **storing data for reuse**, such as a copy of a web page served by a web server. It is cached or stored the first time a user visits the page and the next time a user requests the same page, a cache will serve the copy, which helps keep the origin server from getting overloaded.
- Web caching solutions and strategies enhance page delivery speed significantly and reduce the work needs to be done by the **backend server**.
- Caching servers can be set to refresh at specific intervals or in response to certain events to ensure that the freshest content is cached (useful for rapidly changing information, such as breaking news or rapidly changing pricing).
- Caching can also protect against total outages, delivering already cached content when servers are down.

1.10.1 Proxy Server

GQ. Write short note on Proxy Server.

- An HTTP request from a user is first directed to the network proxy server, or Web cache. Once configured by the network, a browser's request for an object is directed to the Web cache, which must contain updated copies of all objects in its defined **proximity**.
- The main reason for Web caching is to reduce the **response time** for a user request. This benefit is much more obvious when the bandwidth to a requested server is limited because of traffic at certain hours of the day.
- Normally, each organization or **ISP** should have its own cache providing a high-speed link to its users. Consequently, it is to users' advantages that this rapid method of finding objects be available.
- This method of Internet access also reduces traffic on an organization's access link to the Internet. The details of Web caching are as follows:

How to Begin Web Caching Algorithm

1. The source browser makes a TCP connection to the Web cache.
2. The user browser transmits its HTTP request to the Web cache.
3. If it has a copy of the requested object, the Web cache forwards the object to the user browser.

- Otherwise, the Web cache establishes a TCP connection to the requested server and asks for the object. Once it receives the requested object, the Web cache stores a copy of it and forwards another copy to the requesting user browser over the existing TCP connection.

- Fig. 1.10.1 shows three Internet service providers (ISPs). A user in ISP domain 3 is browsing to find and watch an object named http://www.filmmaker.com in ISP domain 1.
- The request for this object is directed to the Web cache, shown by dashed lines. In this example, the Web cache has no record of the requested object and therefore is establishing another TCP connection to update its record.

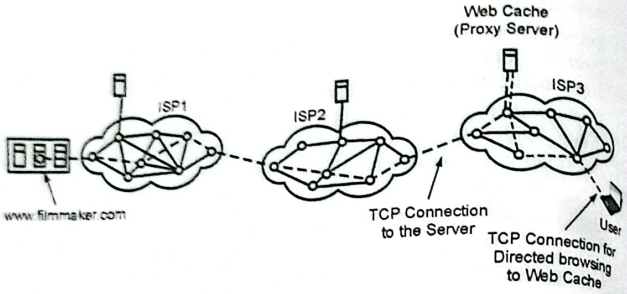


Fig. 1.10.1 : A user's browser requesting an object through theWeb cache

1.10.2 Advantages of Proxy Server

1. It can substantially reduce the response time for repeated requests. (Especially if the bottleneck between the original server and receiver is less than bottleneck between the proxy server and receiver.)
2. It reduces the access link bandwidth (of the university or the office), thereby reducing the cost.
3. It reduces traffic on the Internet as a whole.

1.11 DNS (DOMAIN NAME SYSTEM)

GQ. Explain the need for DNS and describe the protocol functioning.
GQ. Write a short note on DNS.
GQ. Explain DNS in Internet.
UQ. What is DNS? What is server hierarchy?
(SPPU- Q. 4(a), May 17, Q. 3(a), Dec. 19, 4 Marks)
UQ What is DNS Server?
(SPPU- Q. 2(a), May 19, 02 Marks)

- In **application-layer,** protocol defines the way by which applications present on different systems can exchange messages with each other.
- It also defines; the types of messages which are being exchanged, the syntax regarding those message types, the exact meaning of the data, and rules and regulations for the purpose of determining when and how a process forwards as well as responds to messages.

1.11.1 Need for DNS

1. **Web pages, mailboxes** etc. can be access by using the IP addresses of the computers on which they are stored these, but these addresses are hard for people to remember. **DNS servers** eliminate the **need** for humans to memorize IP addresses such as **192.168.66.89**
2. Browsing a company's Web pages from *148.222.98.65* means that if the company moves the Web server to a different machine with a different IP address, then everyone needs to be told the new IP address.
3. Main purpose of high-level, readable names was too introduced to decouple machine names from machine addresses. The network itself understands only numerical addresses so we need some mechanism to convert the names to network addresses.
4. DNS allows **you** to **use** internet more easily by allowing **you** to specify a meaningful name on your web browser instead of using IP address.
5. **Domain Name System (DNS)** is an application layer protocol, which is considered as a name-resolution system, which is very important in the environment of **World Wide Web (WWW).**
6. *DSN is a system, which is considered as responsible for translating fully qualified domain names for example www.kaizenfuturetech.com, into machine-readable IP addresses.*
7. *The Domain Name System enables users to refer alphanumeric names for browsing the WWW, email systems, FTP services, and others, rather than using the systems' Internet Protocol (IP) addresses, which are difficult to remember.*
8. The DNS delegates the task of assigning domain names and mapping these domain names with existing Internet resources. The DNS also mentions the technical functionality regarding the database service, which is at its core.
9. The Domain Name Service defines the DNS protocol, which is considered as a detailed specification of the data structures as well as data communication exchanges referred in the DNS, as part of the Internet Protocol Suite.
10. Two important namespaces are maintained in the internet; the domain name hierarchy and the Internet Protocol (IP) address spaces.
11. The DNS handles the domain name hierarchy and offers translation services in between it and the address spaces. The Internet name servers and a communication protocol implement DNS.
12. A DNS name server maintains the DNS records for a domain; it responds with answers to queries against its database.

1.11.2 Domain

GQ. State and explain different domains in DNS.

- There is a massive number of websites available in the WWW. It is necessary to have a domain or a domain name to make a website visible to the user.
- A domain is a piece of string that helps to identify a website.
- For example, the domain of this website is "kaizenfuturetech.com". The domain name contains an extension to represent the type of the website or the company the website belongs too.
- The .com refers to a global company while .gov represents a government organization. Similarly, .org represents a non-government organization.

There are various kinds of DOMAIN :

1. Generic domain

- It defines the registered hosts according to their generic behaviour.
- Each node in a tree defines the domain name, which is an index to the DNS database.
- It uses three-character labels, and these labels describe the organization type.
- **Example:** .com(commercial) .edu(educational) .mil(military) .org(non profit organization) .net(similar to commercial) all these are generic domain.

2. Country domain

- The format of country domain is same as a generic domain, but it uses two-character country abbreviations (e.g., us for the United States) in place of three-character organizational abbreviations.
- **Example :** .in (india) .us .uk

3. Inverse domain

- The inverse domain is used for mapping an address to a name. When the server has received a request from the client, and the server contains the files of only authorized clients.
- To determine whether the client is on the authorized list or not, it sends a query to the DNS server and ask for mapping an address to the name.
- If we want to know what is the domain name of the website.
- IP to domain name mapping. Therefore, DNS can provide the mapping for example to find the IP addresses of kaizenfuturetech.com then we have to type nslookup www. kaizenfuturetech.com.

(New Syllabus w.e.f academic year 21-22) (P6-55) Tech-Neo Publications ...A SACHIN SHAH Venture

1.11.3 Organization of Domain

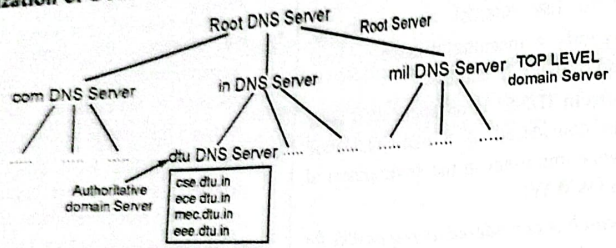


Fig. 1.11.1 : Organization of Domains

- It is Very difficult to find out the IP address associated to a website because there are millions of websites and with all those websites we should be able to generate the IP address immediately, there should not be a lot of delay for that to happen organization of database is very important.
- **DNS record –** Domain name, IP address what is the validity? What is the time to live? and all the information related to that domain name. These records are stored in tree like structure.
- **Namespace –** Set of possible names, flat or hierarchical. Naming system maintains a collection of bindings of names to values – given a name, a resolution mechanism returns the corresponding value –
- **Name server –** It is an implementation of the resolution mechanism. DNS (Domain Name System) = Name service in Internet – Zone is an administrative unit; domain is a subtree.

1.11.4 Components of DNS

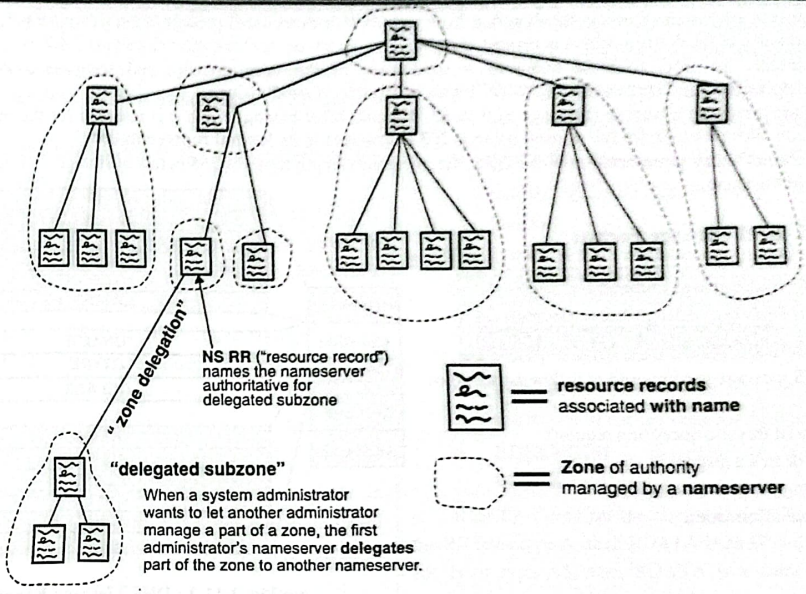
UQ. *List and Explain Component of DNS.* **(SPPU- Q. 3(a), August 15, 06 Marks)**

GQ. *Write components of DNs?*

1. Domain Name Space
2. Name Servers
3. Resolvers

1.11.4(A) Domain Name Space

1. The domain name space is represented as a **tree** data structure. Every node or leaf in the respective tree has a label as well as zero or more **RR (resource records)**, that stores information regarding the domain name.
2. In the domain name there is the label, probably integrated with the parent node's name on the right, separated using a dot. The tree is further sub-divided into several zones starting at the **root zone**.
3. In a DNS zone, there may be a single domain, or many domains with **sub-domains**, based on the administrative preferences of the zone manager.
4. The DNS may also be partitioned based on class in which the independent classes can be considered of as an array of corresponding namespace trees.



(1F1)Fig. 1.11.2 : Domain Name Space

1.11.4(B) Name Servers

1. NAME SERVERS are server **programs**, which **hold information** about the **domain tree's structure** and set information.
2. A name server may cache structure or set information about any part of the domain tree, but in general, a particular name server has complete information about a subset of the domain space, and pointers to other name servers that can be used to lead to information from any part of the domain tree.
3. Name servers know the parts of the domain tree for which they have complete information; a name server is said to be an **AUTHORITY** for these parts of the name space.
4. Authoritative information is organized into units called **ZONEs**, and these zones can be automatically distributed to the name servers, which provide redundant service for the data in a zone.

1.11.4(C) Resolvers

1. RESOLVERS are **programs** that extract information from name servers in response to client requests.
2. Resolvers must be able to access at least one name server and use that name server's information to answer a query directly, or pursue the query using referrals to other name servers.
3. A resolver will typically be a system routine that is directly accessible to user programs; hence, no protocol is necessary between the resolver and the user program.

Domain Name Syntax

GQ Write note on Domain Name Syntax.

1. In a domain name there may be one or more parts, which are technically referred as **labels**. These labels are typically concatenated, and delimited by the use of dots, for example **kaizenfuturetech.com**.
2. The label present at right-most side conveys the top-level domain; for example, the domain name www.example.com belongs to the top-level domain com.

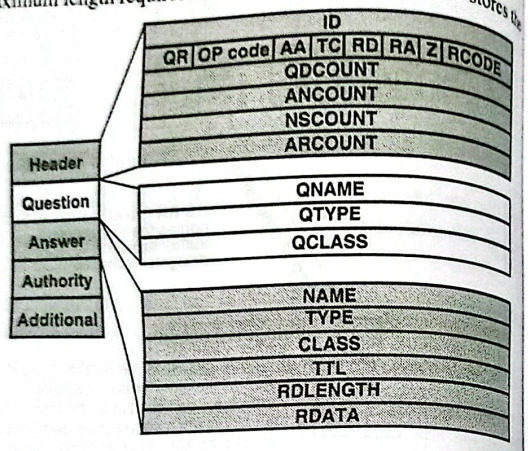
3. The hierarchy of domains flows in the direction from right to left, every label present at the left indicates a subdivision or sub-domain of the domain, which is present at right.
4. For example, the label example mentions a subdomain of the com domain, and www is a subdomain of kaizenfuturetech.com. There may be up to **127 levels** in this tree of subdivisions.
5. Characters present in a label in the range of **0 to 63**. The null label having length 0 is reserved for the root zone. The maximum allowed length for full domain name is **253 characters** in its **textual representation.**
6. In the internal binary representation of the DNS, the maximum length requires **255 octets** of storage, as it also stores the length of the name.

1.11.5 DNS Message Format

GQ. Explain DNS message format.
UQ. Explain DNS Request and Response message format. **(SPPU- Q. 1(a), Dec. 15, 06 Marks)**

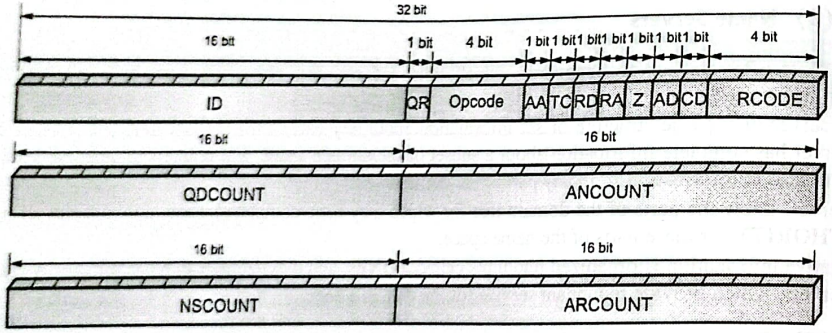
The DNS message is composed of following sections:

1. Header
2. Question (if this is a query or a request)
3. Answer (if this a response)
4. Authority
5. Additional information



(1F2)**Fig. 1.11.3 : DNS Message Format**

DNS Message Format : The Header Section



(1F3)**Fig. 1.11.4 : Header Section of DNS Message**

- The DNS message header is 12B long (32*3/8).
  - **ID** : Also referred as **Identification** or **Transaction ID**. It identifies a query-response pair in a DNS communication from QR to RCODE, these are **flags**.
  - **QDCOUNT** : The number of questions in the message,
  - **ANCOUNT** : The number of RRs in the answer in the message,
  - **NSCOUNT** : The number of authority RRs in the message,
  - **ARCOUNT** : The number of additional information RR in the message.
- Here are some of the flags that appear in the DNS message header (whether the message is a query or a response) :
  - **QR :** Identifies whether this message is a query or a response.
  - **Opcode :** If this value is 0 then the message is a standard query.

  - **Authoritative Answer (AA) :** This bit is available only for DNS replies. It tells whether the server is authoritative for the requested domain name.
- AA = 1; the DNS server is authoritative on the domain name
- AA = 0; the DNS server is not authoritative on the domain name
  - **Truncated (TC) :** Tells whether the message is truncated or not. A DNS message is truncated when it cannot fit in a single UDP datagram with a maximum size of 512 Bytes.
  - **Recursion Desired (RD) :** Expresses the querying host's desire to make a recursive query or not.
- If set to one, then it means, "The querying host desires a recursive query". This flag is copied in the response too.
  - **Recursion Available (RA) :** Appears only in DNS responses. It tells whether the name server that receives the query can do recursive queries or not
- If set to 1, this flag says "I can do recursive queries", If set to 0, this flag says "Sorry dude I cannot do recursive queries".
- **Zero:** Actually, set to 0. This flag is developed for future use.
  - **Reply CODE (RC) :** This flag is only valid in a DNS reply. It tells whether the response contains errors or not
- if RCODE = 0, then there are no errors, else : there is an error

DNS message : the remaining sections

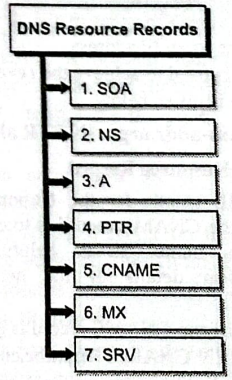
- We learned the parts that constitute the Header. This is just one part of a DNS message. In fact, the DNS message contains, in addition to the header, other sections :
- **Question section** -if it is a DNS query or a reply. Strangely enough, the DNS reply contains the Question section too, **Answer section**- if it's a DNS reply,
- **The Question section in a DNS message :** When the DNS message is a DNS query, then it has a header and a Question section. The Question section is composed of the **three 16-bits fields** :
  - **QNAME :** QNAME is the domain name encoded in labels (we will learn about labels later),
  - **QTYPE:** determines the type of the query. You can find a complete list of types on Wikipedia,
  - **QCLASS:** determines the class of the query. Usually QCLASS has the value "IN" to mean "Internet".
- **The Answer section contains the following parts:**
  - **NAME :** has the same format as QNAME.
  - **CLASS :** has the same format as QCLASS.
  - **TYPE :** has the same format as QTYPE.
  - **TTL :** describes how much time - in seconds - can this record be cached before it must be discarded.
  - **RDLENGTH :** describes the length of the RDATA field.

RDATA : contains the resource itself. For example, if the RR is of type A, then RDATA is an IPv4 address. If the RR is of type NS, then RDATA is a name server alias hostname

1.11.6 Resource Record and Types of Name Server

GQ. Explain its various resource records with one example.

- **DNS resource records** are contents of the DNS zone file. The zone file contains mappings between domain names and IP addresses in the form of text records.
- There are many types of the resource records.



(1F4)**Fig. 1.11.5 : DNS resource records**

1. SOA Record

(i) Every zone file will have a SOA record. It will be present at the beginning of the zone. The SOA stands for Start of Authority. Normally, this type of record holds information about the zone itself and about other records. Each zone will be having only one SOA record.

(ii) The SOA record contains the following fields.

e.g.: IN SOA

nameserver.place.dom. postmaster.place.dom.

2. NS Record

(i) The NS record stands for nameserver record. They indicate primary and secondary servers for the zone specified in the SOA record. Zones can contain many NS records, but it should contain at least one NS record for a DNS zone.

(ii) **For example**; when the administrator on abc.com delegated authority for the noam.abc.com subdomain to noamdc1.noam.abc.com., the following line was added to the zone abc.com and noam.abc.com:

noam.abc.com. IN NS noamdc1.noam.abc.com.

3. A Record

(i) The A record stands for Address record. It maps a domain name to an IP address so that the resolver can request the corresponding IP address for the domain.

(ii) As an example, the following a resource record, located in the zone abc.com, maps the **FQDN** of the server to its IP address.

abc.com IN A 172.16.48.1

4. PTR Records

(i) The PTR record stands for the pointer record. It functions reversely as that of the A record. It maps a domain name to an IP address.

(ii) This record is used to achieve the **reverse dns.**

(iii) An example is given below.

1.48.16.172.in-addr.arpa. IN PTR abc.com.

5. CNAME Resource Record

(i) The CNAME stands for the **Canonical name.** The function of the CNAME record is to create an alias for the domain name. It is helpful to hide the implementation details of the network from the customers.

(ii) An example of the CNAME record is given below.

ftp.abc.com. IN CNAME ftp1.abc.com.

(iii) Once a DNS client queries for the A resource record for ftp.abc.com, the DNS server finds the CNAME resource record.

(iv) Then it resolves the query for the A resource record for ftp1.abc.com, and returns both the A and CNAME resource records to the client. This is how CNAME record works.

6. MX Resource Records

(i) The MX record stands for the **mail exchange record.** The mail exchange (MX) resource record specifies a mail exchange server for a DNS domain name.

(ii) A mail exchange server is a host that will either process or forward mail for the DNS domain name. Processing the mail means either delivering it to the addressee or passing it to a different type of mail transport.

(iii) Forwarding the mail means sending it to its final destination server. Only mail exchange servers use MX records. We can have multiple **MX** resource records for that domain.

(iv) The following example shows **MX** resource records for the mail servers for the domain **noam.abc.com. :**

***. noam.abc.com. IN MX 0**

mailserver1.noam.abc.com.

***. noam.abc.com. IN MX 10**

mailserver2.noam.abc.com.

***. noam.abc.com. IN MX 10**

mailserver3.noam.abc.com.

7. SRV Records

(i) With MX records, we can have multiple mail servers in a DNS domain, and when a mailer needs to send mail to a host in the domain, it can find the location of a mail exchange server.

(ii) Service (SRV) resource records enable user to specify the location of the servers for a specific service, protocol, and DNS domain.

(iii) Thus, if user has two Web servers in his domain, he can create SRV resource records specifying which hosts serve as Web servers, and resolvers can then retrieve all the SRV resource records for the Web servers.

1.11.7 Types of Name Server

UQ. List and Explain different types of name Servers. **(SPPU- Q. 3(b), August 14, 04 Marks)**

1. Recursive resolvers
2. Root name servers
3. Top level domain mail servers
4. Adhortative name servers

1. Recursive resolvers : A **recursive DNS** lookup is where one **DNS server** communicates with several other **DNS servers** to hunt down an IP address and return it to the client. This is in contrast to an iterative **DNS** query, where the client communicates directly with each **DNS server** involved in the lookup.

2. Root name servers : A **root name server** is a **name server** for the **root** zone of the Domain Name System (DNS) of the Internet. It directly answers requests for records in the **root** zone and answers other requests by returning a list of the authoritative **name servers** for the appropriate **top-level domain (TLD).** There are 13 root name servers.

3. Top Level Domain (TLD) name servers: A TLD name server maintains information for all the domain names that share a common domain extension, such as .com, net, or whatever comes after the last dot in a url. For example, a .com **TLD nameserver** contains information for every website that ends in **'.com'.**

4. Authoritative name servers : An authoritative Nameserver is a nameserver **(DNS Server)** that holds the actual DNS records (A, CNAME, PTR, etc) for a particular domain/ address. Authoritative DNS nameservers are responsible for providing answers to recursive DNS nameservers about where specific websites can be found. These answers contain important information for each domain, like IP addresses.

1.11.7(A) Root Servers

1. It 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.
2. There are several root servers, each covering the whole domain name space. The root servers are distributed all around the world.

1.11.7(B) Primary and Secondary Servers

1. **A primary server :** loads all information from the disk file; **Primary DNS servers** contain all relevant resource records and handle **DNS** queries for a domain. A **primary server** is a **server** that acts as the first source for Domain Name System (DNS) data and responds to queries. It can be contrasted to the secondary **server**, which acts like the **primary server** but does not have the same access to data.
2. **The secondary server:** loads all information from the primary server. **Secondary DNS servers** contain zone file copies that are read-only, meaning they cannot be modified.

1.11.8 Domain Resolution Definition

- Domain Resolution is a service that points the domain name to the website space IP so that people can easily access the website through the registered domain name. An IP address is a numeric address that identifies a site on the network.
- In order to facilitate the memory, the domain name is used instead of the IP address to identify the site address. Domain resolution is the process of converting domain names to IP addresses. The resolution of the domain name is done by the DNS server.
- Domain resolution is also called domain pointing, server settings, domain configuration, reverse IP registration, and so on. To put it simply, the easy-to-remember domain name is resolved into IP.
- The service is completed by the DNS server, which resolves the domain name to an IP address, and then binds a subdirectory to the domain name on the host of this IP address.
- The address on the Internet is a digital IP address, and the main purpose of domain resolution is to facilitate memory.

1.11.8(A) How Domain Resolution Works

UQ. Describe the process of name resolution in DNS. **(SPPU- Q. 5(a), August 14, 05 Marks)**

UQ. Explain domain name resolution process. **(SPPU- Q. 4(a), May 17, 03 Marks, Q. 3(a), Dec. 19, 6 Marks)**

- After registering a domain name with a domain registrar, how can I see the content of my website? In a professional term, it is called "domain name resolution".
- The domain name is a set of address conversion systems specially established for the convenience of memory. To access a server on the Internet, it must ultimately be achieved through an IP address. Domain name resolution is the process of converting a domain name into an IP address.
- One domain name corresponds to one IP address, and one IP address can correspond to multiple domain names; therefore, multiple domain names can be resolved to one IP address at the same time. Domain

name resolution needs to be completed by a dedicated domain name resolution server (DNS).

1.11.8(B) DNS Lookup

UQ. Explain lookup methods used by the DNS to resolve the remote names.
(SPPU- Q. 2(a), May 19, 04 Marks)

- For most situations, DNS is concerned with a domain name being translated into the appropriate IP address.
- To learn how this process works, it helps to follow the path of a DNS lookup as it travels from a web browser, through the DNS lookup process, and back again.
- Note: Often DNS lookup information will be cached either locally inside the querying computer or remotely in the DNS infrastructure.
- There are typically 8 steps in a DNS lookup. When DNS information is cached, steps are skipped from the DNS lookup process which makes it quicker. The example below outlines all 8 steps when nothing is cached.

The 8 steps in a DNS lookup:

1. A user types 'example.com' into a web browser and the query travels into the Internet and is received by a DNS recursive resolver.
2. The resolver then queries a DNS root nameserver (.).
3. The root server then responds to the resolver with the address of a Top Level Domain (TLD) DNS server (such as .com or .net), which stores the information for its domains. When searching for example.com, our request is pointed toward the .com TLD.
4. The resolver then makes a request to the .com TLD.
5. The TLD server then responds with the IP address of the domain's nameserver, example.com.
6. Lastly, the recursive resolver sends a query to the domain's nameserver.
7. The IP address for example.com is then returned to the resolver from the nameserver.
8. The DNS resolver then responds to the web browser with the IP address of the domain requested initially.

Once the 8 steps of the DNS lookup have returned the IP address for example.com, the browser is able to make the request for the web page:

9. The browser makes a HTTP request to the IP address.
10. The server at that IP returns the webpage to be rendered in the browser (step 10).

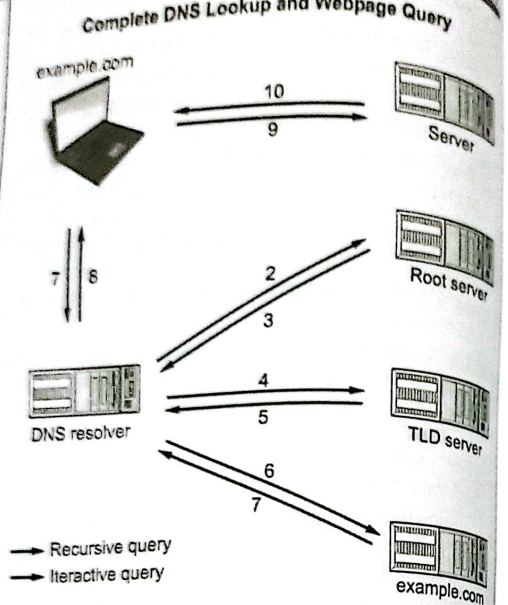


Fig. 1.11.6 : DNS Look Up and Webpage Query

1.11.8(C) DNS resolver

UQ. Describe the process of name resolution in DNS.
(SPPU- Q. 5(a), August 14, 05 Marks)

UQ. State and explain. Name Address resolution techniques in DNS.
(SPPU- Q. 4(a), May 16, 06 Marks)

- The DNS resolver is the first stop in the DNS lookup and it is responsible for dealing with the client that made the initial request.
- The resolver starts the sequence of queries that ultimately leads to a URL being translated into the necessary IP address.
- **Note:** A typical uncached DNS lookup will involve both recursive and iterative queries.
- It's important to differentiate between a recursive DNS query and a recursive DNS resolver. The query refers to the request made to a DNS resolver requiring the resolution of the query. A DNS recursive resolver is the computer that accepts a recursive query and processes the response by making the necessary requests.

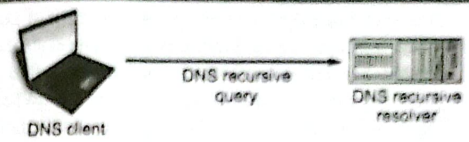


Fig. 1.11.7 : DNS Query

1.11.8(D) Types of DNS Queries

UQ. What are the query resolution techniques in DNS? Explain any one of them.
(SPPU- Q. 6(a), August 17, 04 Marks)

- In a typical DNS lookup three types of queries occur. By using a combination of these queries, an optimized process for DNS resolution can result in a reduction of distance traveled.
- In an ideal situation cached record data will be available, allowing a DNS name server to return a non-recursive query.

Three types of DNS queries

1. **Recursive query** - In a recursive query, a DNS client requires that a DNS server (typically a DNS recursive resolver) will respond to the client with either the requested resource record or an error message if the resolver can't find the record.
2. **Iterative query** - in this situation the DNS client will allow a DNS server to return the best answer it can. If the queried DNS server does not have a match for the query name, it will return a referral to a DNS server authoritative for a lower level of the domain namespace. The DNS client will then make a query to the referral address. This process continues with additional DNS servers down the query chain until either an error or timeout occurs.
3. **Non-recursive query** - typically this will occur when a DNS resolver client queries a DNS server for a record that it has access to either because it's authoritative for the record or the record exists inside of its cache. Typically, a DNS server will cache DNS records to prevent additional bandwidth consumption and load on upstream servers.

1.12 BOOTP (BOOTSTRAP PROTOCOL)

GQ. Explain Bootstrap Protocol ?

Bootstrap Process

It is a method of accessing the information of an internet connected computer such as (**IP address, subnet mask, router address, IP address of the name server**) stored in a configuration file these pieces for information are need be known to the computer connected to a TCP/IP internet.

Bootstrap Protocol (BOOTP)

- It is a client-server protocol designed to obtain the above given information (**i.e., IP address, subnet mask, router address, IP address of the name server**) from a diskless computer or a computer booted for the first time.
- The operating system and networking software are stored in the **read-only memory (ROM)**, if the computer or workstation is diskless.
- BOOTP is implemented using the User Datagram Protocol (UDP) as transport protocol, **port number 67** is used by the server to receive client requests and **port number 68** is used by the client to receive server responses. BOOTP operates only on IPv4 networks.
- BOOTP was originally defined in 1985 for Request for Comments (RFC) 951 and 1084 to replace Reverse Address Resolution Protocol, which required servers to be present on each server IP address. With BOOTP, a central BOOTP server could exist for numerous subnets.
- Historically, BOOTP has also been used for Unix-like diskless workstations to obtain the network location of their boot image, in addition to the IP address assignment. Enterprises used it to roll out a pre-configured client (**e.g., Windows**) installation to newly installed PCs.
- BOOTP is the basis for Dynamic Host Configuration Protocol (DHCP). DHCP servers are used to receive client requests.
- ARP is the predecessor of BOOTP and serves the same purpose, but the limitation of the RARP is that it only provides the information about IP not the extra information regarding that.

Limitation of RARP

1. Networks which do not have a broadcast method can't support RARP as it uses the broadcast method of the MAC layer underneath the IP layer.
2. RARP is heavily dependent on the MAC protocol.
3. RARP just supplies the IP address corresponding to a MAC address It doesn't support respond with any more data.

4. RARP uses the computer hardware's address to identify the machine and hence cannot be used in networks that dynamically assign hardware addresses.

1.12.1 Working of BOOTP

GQ. Explain Working of BOOTP

- When a BOOTP client is started, it has no IP address, so it broadcasts a message containing its MAC address onto the network. This message is called a "**BOOTP request**," and it is picked up by the BOOTP server, which replies to the client with the following **information that the client needs:**
  1. The client's IP address, subnet mask, and default gateway address
  2. The IP address and host name of the BOOTP server
  3. The IP address of the server that has the boot image, which the client needs to load its operating system
- The host sends a BOOTP request and uses **UDP source port 68** and **destination port 67**. This packet is a broadcasted so everything in the broadcast domain receives it. On our network, we have a BOOTP server listening on **UDP port 67**

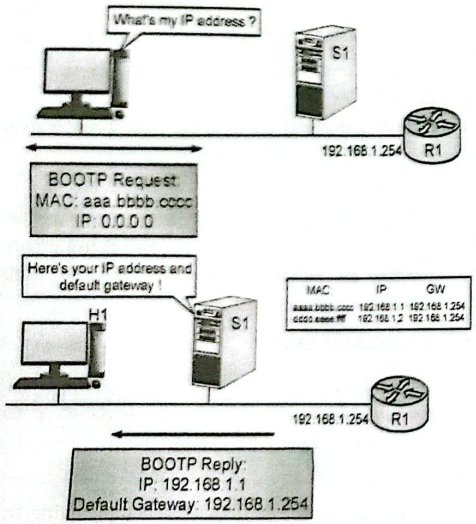


Fig. 1.12.1 : Working of BOOTP

- The server sees the broadcast packet from the host and since it is listening on **UDP port 67,** it processes the packet. The server then looks in its database to find a matching entry for the MAC address of the host. When there is a match, it returns the information to the host with a unicast packet
- When the client receives this information from the BOOTP server, it configures and initializes its TCP/IP protocol stack, and then connects to the server on which the boot image is shared. The client loads the boot image and uses this information to load and start its operating system.

1.12.2 Different Scenarios of Operation

Enterprises can use BOOTP in two different ways, with the client and server on the same network or with the client and server on different networks.

1.12.2(A) Scenario 1: Client and server on the Same Network

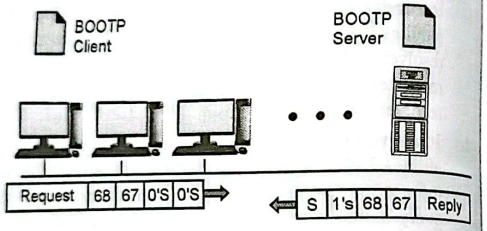


Fig. 1.12.2 : Client and Server on the same Network

- In this scenario, when the BOOTP client is booted up, it initially has no IP configuration. Instead, it broadcasts its media access control address to the network as a BOOTP request.

  The BOOTP server then picks up the message and replies with the following:
  - IP address from the domain name system;
  - hostname;
  - subnet mask;
  - default gateway address;
  - boot file; and
  - transaction ID.
- Once the client receives this information, it initializes Transmission Control Protocol/Internet Protocol and connects to the receiving server where the boot image is located.

1.12.2(B) Scenario 2 : Client and Server on Different Networks

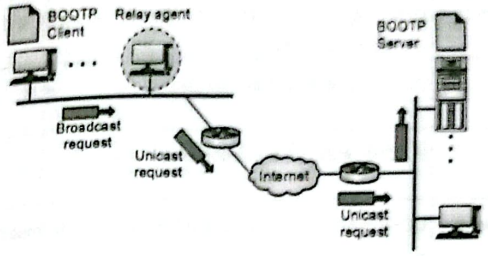


Fig. 1.12.3 : BOOTP client and server on the same and different networks

- In this scenario, the BOOTP request's broadcast IP datagram can't pass through the router. To resolve this, a **vendor extension** is used as an intermediary to act as the relay agent.
- The relay agent is configured with the **unicast address.** When the agent receives the broadcast packet, it sends the unicast diagram to the BOOTP server. This is then routed to the BOOTP client.

1.12.3 Configuration Information

The important information's provided are :

- IP address
- IP address of the default router for that particular subnet
- Subnet mask
- IP addresses of the primary and secondary nameservers

Additionally it may also provide :

- Time offset from GMT
- The IP address of a time server
- The IP address of a boot server
- The name of a boot file (e.g. boot image for X terminals)
- The IP domain name for the client

1.12.4 Difference between BOOTP and DHCP

GQ. State difference between BOOTP and DHCP.

Table 1.12.1 : Comparison of BOOTP and DHCP

Criteria for Comparison	BOOTP	DHCP
Acronym For	BOOTP stands **for Bootstrap Protocol.**	DHCP stands for **Dynamic host configuration protocol.**
IP	BOOTP can only provide an IP to a computer while it is booting.	DCHP can provide an IP when the OS is already loaded.
Client Configuration	BOOTP supports a **limited number of client** configuration parameters referred to as vendor extensions.	DHCP supports a **larger and extensible set of client** configuration parameters referred to as options.
System Restart	BOOTP client **do not rebind or renew configuration** with the BOOTP server except when the system restarts.	DHCP clients **do not require a system restart to rebind** or renew configuration with the DHCP server.
Configuration Process	BOOTP uses a **two-phase bootstrap configuration** process in which clients contact BOOTP servers to perform address determination and boot file name selection and clients contact Trivial File Transfer Protocol (TFTP) servers to perform file transfer of their boot image.	DHCP uses a **single-phase boot configuration** process whereby a DHCP client negotiates with a DHCP server to determine its IP address and obtain any other initial configuration details it needs for network operation.

Criteria for Comparison	BOOTP	DHCP
Lease	BOOTP has a **30-day lease** on the IP address as a default.	DHCP has **eight-day lease duration** for Microsoft and one day for Cisco routers.
Mobile Machine Support	BOOTP **does not** support mobile machines.	DHCP **supports** mobile machines.
Configuration Type	In BOOTP, **manual-configuration** takes place.	In DCHP, **auto-configuration** takes place.
IP Addressing	BOOTP **does not provide** temporary IP addressing.	DCHP **provides** temporary IP addressing for only limited amount of time.
Possibility Of Errors	Due to **manual-configuration**, BOOTP is faced with errors.	Due to **auto-configuration** in DHCP, it is immune to errors.
Storage Disk	BOOTP provides the **information to the diskless** computer or workstation.	DCHP **requires disks** to store and forward the information.
Compatibility With Clients	BOOTP is **not compatible** with DHCP clients.	DHCP **support** BOOTP clients.

1.13 FTP (FILE TRANSFER PROTOCOL)

GQ. What is the purpose of FTP ? What are the three FTP transmission modes ? Explain.
GQ. Explain FTP in detail with respect to server and client communication.
GQ. Describe the functions of the FTP connections. Explain what kinds of file types FTP can transfer.
UQ. What is FTP? Where and when it is used? Why does it require 2 ports ? **(SPPU- Q. 6(a), Aug. 15, Q. 5(a), Aug. 17, 06 Marks, Q. 5(b), Oct. 16, 02 Marks)**

1. The File Transfer Protocol (FTP) is a standard network protocol used for the **transfer of computer files** between a client and server on a computer network.
2. FTP is built on **client-server model** architecture and uses separate control and data connections between the client and the server.
3. FTP users may authenticate themselves with **a clear-text sign-in protocol**, normally in the form of a username and password, but can connect anonymously if the server is configured to allow it.
4. The first FTP client applications were command-line programs developed before operating systems had graphical user interfaces, and are still shipped with most **Windows, Unix, and Linux operating systems**.
5. Many FTP clients and automation utilities have since been developed for desktops, servers, mobile devices, and hardware, and FTP has been incorporated into productivity applications, such as web page editors.
6. FTP may run in ***active* or *passive* mode**, which determines how the data connection is established. In both cases, the client creates a TCP control connection from a random, usually an unprivileged, port N to the **FTP** server command **port 21**.
7. In active mode, the client starts listening for incoming data connections from the server on port M. It sends the **FTP** command **PORT M** to inform the server on which port it is listening.
8. The server then initiates a data channel to the client from its **port 20**, the **FTP** server data port. In situations where the client is behind a firewall and unable to accept incoming **TCP** connections, passive mode may be used.
9. In this mode, the client uses the control connection to send a **PASV** command to the server and then receives a server IP address and server port number from the server, which the client then uses to open a data connection from an arbitrary client port to the server IP address and server port number received.

1.13.1 FTP Login

1. FTP login uses normal username and password scheme for granting access. The username is sent to the server using the **USER** command, and the password is sent using the **PASS** command.
2. This sequence is unencrypted **"on the wire"**, so may be defenceless in case of network sniffing attack. If the server accepts the information provided by the client, the server will send a greeting to the client and the session will commence.
3. If the server supports it, users may log in without providing login credentials, but the same server may authorize only limited access for such sessions.

1.13.2 FTP Commands

GQ. List and Explain various FTP commands ?
UQ. Explain any 4 commands used in FTP. **(SPPU- Q. 5(b), Oct. 16, 05 Marks)**
UQ. State and explain six commands in FTP. **(SPPU- Q. 4(a), Dec. 16, 06 Marks)**
UQ. Why it faster? Explain at least 3 user commands used in FTP. **(SPPU- Q. 5(a), August 17, 06 Marks)**

Following are some important FTP commands :

Table 1.13.1 : FTP Commands

Command	Description
?	To request help or information about the FTP commands
Ascii	To set the mode of file transfer to ASCII (this is the default and transmits seven bits per character)
Binary	To set the mode of file transfer to binary (the binary mode transmits all eight bits per byte and thus provides less chance of a transmission error and must be used to transmit files other than ASCII files)
Bye	To exit the FTP environment (same as quit)
Cd	To change directory on the remote machine
Close	To terminate a connection with another computer
close Brubeck	Closes the current FTP connection with brubeck, but still leaves you within the FTP environment.
Delete	To delete (remove) a file in the current remote directory (same as rm in UNIX)
Get	To copy one file from the remote machine to the local machine
close Brubeck	Copies file ABC in the current remote directory to (or on top of) a file named DEF in your current local directory.
get ABC	Copies file ABC in the current remote directory to (or on top of) a file with the same name, ABC, in your current local directory.
Help	To request a list of all available FTP commands
Lcd	To change directory on your local machine (same as UNIX cd)
Ls	To list the names of the files in the current remote directory
Mkdir	To make a new directory within the current remote directory

Command	Description
Mget	To copy multiple files from the remote machine to the local machine; you are prompted for a y/n answer before transferring each file
mget *	Copies all the files in the current remote directory to your current local directory, using the same filenames. Notice the use of the wild card character(*).
Mput	To copy multiple files from the local machine to the remote machine; you are prompted for a y/n answer before transferring each file
Open	To open a connection with another computer
open Brubeck	Opens a new FTP connection with brubeck: You must enter a username and password for a Brubeck account (unless it is to be an anonymous connection).
Put	To copy one file from the local machine to the remote machine
Pwd	To find out the pathname of the current directory on the remote machine
Quit	To exit the FTP environment (same as bye)
Rmdir	To remove (delete) a directory in the current remote directory

1.13.3 FTP Transmission Modes

The following transmission modes are defined in FTP:

1. **Stream mode :** Data is sent as a continuous stream, relieving FTP from doing any processing. Rather, all processing is left up to TCP. No End-of-file indicator is needed, unless the data is divided into records.
2. **Block mode :** FTP breaks the data into several blocks (block header, byte count, and data field) and then passes it on to TCP.
3. **Compressed mode :** Data is compressed using a simple algorithm (usually run-length encoding).

1.13.4 File Transfer with FTP

GQ. Can we specify the transfer in a Web page ? Explain with the help of suitable example.

It is possible to transfer file with FTP. Following are steps to transfer file with FTP :

1. On the command line, enter FTP <server name>.
2. Enter your login information if prompted.
3. Set your transfer mode to either 'ascii' or 'binary' depending upon the type of file you are transferring.
4. You can discover what directory you have connected to by entering the command 'pwd'.
5. To change directories on the remote machine, enter 'cd' and the name of the directory.
6. To change directories locally, enter 'lcd' To put a file on the remote machine, enter PUT and the name of the file.
7. Once the transfer completes, you can enter 'close' and then 'quit' ('!' and 'bye' also serve the same function as quit).

1.13.5 Differentiate between HTTP and FTP

UQ. Differentiate between HTTP and FTP **(SPPU- Q. 3(a), Dec. 14, 04 Marks)**

Table 1.13.2 : Comparison of HTTP and FTP

Parameter	HTTP	FTP
Functionality	Basic functionality is to access websites.	FTP transfers files from one host to another.
Connection	Only data connection is established.	FTP establishes two connections: one for data and one for the control connection.
TCP ports	HTTP uses TCP's port number 80.	FTP uses TCP's port number 20 and 21.
Efficiency	HTTP is efficient in the process of transferring smaller files like web pages.	FTP is efficient in the process of transferring larger files.
Authentication	No need of authentication.	FTP requires a password.
Data	The content transferred to a device using HTTP is not saved to the memory of that device.	The file transferred to the host device using FTP is saved in the memory of that host device.

1.14 TFTP (TRIVIAL FILE TRANSFER PROTOCOL)

GQ. Write short note of TFTP.

- Trivial File Transfer Protocol is a simple protocol that is used for sending a file from the server to the client.
- TFTP uses the concept of UDP (User Datagram Protocol) to share files between server and client.
- TFTP has a very simple concept, and due to its simple concept, it has a straightforward design too.
- In general, TFTP does not follow any authentication before the communication of file.
- TFTP does not apply any security mechanism while filing communication. Since Trivial File Transfer Protocol does not follow any authentication mechanism or any security mechanism, it could not be used over the internet to communicate files.
- It is generally used for communicating files among machines set up in the local intranet only.
- TFTP's most important feature is that it uses a minimal amount of memory. TFTP could be used to communicate boot files if computers do not have hard disks. TFTP generally uses protocol 69, however, the port used for communication could be defined by used when Trivial File Transfer Protocol is being set up.

1.14.1 Features TFTP

GQ. List features of FTP.

1. TFTP is based on client/server principle.
2. It uses Well-known UDP port number 69 for TFTP server.
3. TFTP is unsecured protocol.
4. TFTP does not support authentication.
5. Every TFTP data unit has a sequence number.
6. Each data unit is individually acknowledged. After receiving the acknowledgement, the next data unit is sent.
7. Error recovery is by retransmission after timeout.

1.14.2 Use of TFTP

- TFTP is used for communicating files between client and server within the local network.
- TFTP is beneficial when the client computer has very low memory storage devices or hard disk devices.
- It could be used to communicate boot files when the server is on the client's computer.
- Since it is easier to implement; hence, it could be widely used when we have a low-security mechanism to be followed.
- It does not follow any authentication mechanism; hence only those files could be communicated, which does not need to have any security mechanism.

1.14.3 Working of TFTP

UQ. Describe working of TFTP. **(SPPU- Q. 5(a), Dec. 14, 10 Marks)**

Let understand the mechanism of how communication takes place between a client and a server.

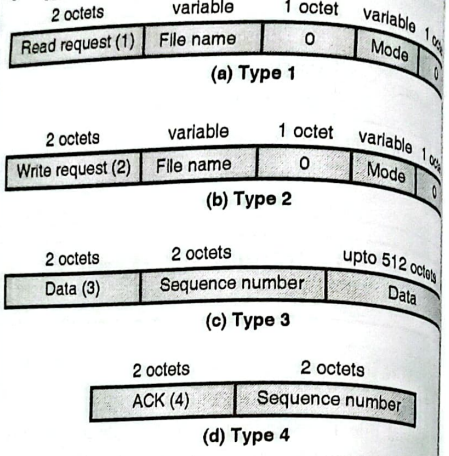
- Since TFTP uses UDP for communicating files, hence it establishes a connection generally by using port 69.
- Once the connection is established, the client generally requests RRQ (Read Request) or WRQ (Write Request).
- A client generally requests for reading requests if it wants only to read the file and generated a written request if he wants to write a particular file that exists on the server.
- Once this is done, then files are communicated in the form of small packets.
- In either case, the server selects a UDP port to be used for further dialogue and sends its first response to the client through the selected UPD port.
- These packets are 512 bytes. The file to be communicated is divided into small packets, where each packet consists of 512 bytes.
- Once a packet is communicated from server to client, the server waits to receive an acknowledgement from the client that the packet has been received.
- Once the acknowledgement is received, the server sends the next packet of 512 bytes.
- This is done till the last packet is communicated from server-side to client-side.
- The last data block containing EDF or a data block containing less than 512 octets terminates the session.
- Error recovery is done using retransmission after timeout.
- If TFTP message is lost and if there is no expected response, the message is repeated by the sender after time out.
- If the next data message is not received after acknowledgement, the last acknowledgement is repeated after timeout.
- The last packet which is generated for sharing a particular file is always less than 512 bytes.
- Even if the packets generated are in multiples of 512 bytes, then it sends an additional packet which is less than 512 bytes so that the client could understand that it has received the file.

1.14.4 Types and TFTP Message Formats

GQ. List and Describe type of TFTP Message Formats.

There are four types of TFTP messages :

- Read request
- Write request
- Data Packets
- Acknowledgement request



(1A14)**Fig. 1.14.1 : Types of TFTP message**

The first two octets indicate the type of message. Mo field defines the type of data (ASCII, binary, Mail). T filename and mode fields are delimited using an all zero octet.

1. **Read request (Type 1).** This is used by the client get a copy of a file from the server.
2. **Write request (Type 2).** This command is used by t client to write a file into the server.
3. **Data Packet (Type 3).** this command contains block data (portion of the file being copied). This messa contains the data block of fixed size of 512 octets. T session is terminated if a data message arrives with da octet less than 512 octets.
4. **Acknowledgement (Type 4).** The last data messa can have data block with EOF having size less than 51 octets. This is used by the client and the server acknowledge the received data units.

1.14.5 Difference between FTP and TFTP

GQ. State difference between FTP and TFTP.

UQ. Differentiate FTP and TFTP. **(SPPU- Q. 4(a), Dec. 14, 04 Marks)**

Sr. No.	FTP	TFTP
1.	FTP stands for File Transfer Protocol.	TFTP stands for Trivial File Transfer Protocol.
2.	The software of FTP is larger than TFTP.	While software of TFTP is smaller than FTP.
3.	FTP works on two ports: 20 and 21.	While TFTP works on 69 Port number.
4.	FTP services are provided by TCP.	While TFTP services are provided by UDP.
5.	The complexity of FTP is higher than TFTP.	While the complexity of TFTP is less than FTP complexity.
6.	There are many commands or messages in FTP.	There are only 5 messages in TFTP.
7.	FTP need authentication for communication.	While TFTP does not need authentication for communication.
8.	FTP is generally suited for uploading and downloading of files by remote users.	While TFTP is mainly used for transmission of configurations to and from network devices.
9.	FTP is a reliable transfer protocol.	While; TFTP is an unreliable transfer protocol.
10.	FTP is based on TCP.	While; TFTP is based on UDP.
11.	FTP is slower.	TFTP is faster as compared to FTP.

1.14.6 Advantages and Disadvantages of TFTP

Advantages

1. It is a fast file transfer protocol.
2. Network device configuration files can be easily transferred with this protocol.
3. It can be easily used with 3rd party software on windows and linux operating system.
4. Where it is not necessary to sued FTP. It is recommended to use this product.
5. Uses UDP as the transport protocol (unlike FTP using TCP port 21).
6. Used to read or Write files from the remote server.
7. It supports three different transmission mode "netascii","octet","mail".

Disadvantages

1. It is an unsecured file transfer protocol.
2. It does not provide file security compared to FTP.
3. It performs file transfer and file transfer only.
4. There is no authentication or encryption mechanism.
5. It does not list the content of directories.

1.15 ELECTRONIC MAIL

GQ. Write short note on electronic mail system.

- One of the most popular network services is electronic mail (e-mail).
- **Simple Mail Transfer Protocol (SMTP)** is the standard mechanism for electronic mail on the internet. The first e-mail systems simply consisted by file transfer protocols.

Limitations of this system were as follows :

1. It is difficult to send a message to a group of people.
2. Message did not have any internal structure. Therefore, its computer processing was difficult.
3. The sender never used to know if a message arrived or not.
4. It was not easy to handover one's e-mails to someone else for managing them when one is out of town or country for some time.
5. The user interface with the transmission system is poorly integrated.

6. It was not possible to create and send messages containing a text, drawing, facsimile and voice together.

- So more elaborate e-mail systems were proposed. **ARPANET e-mail** proposals were published as RFC 821 (transmission protocol) and RFC 822 (message format). These are used on Internet.

1.15.1 E-mail Architecture and Services

UQ. Explain email architecture & its services. **(SPPU- Q. 3(b), May 15, 04 Marks)**

An e-mail system consists of two subsystems :

1. User agents and 2. Message transfer agents

- **User agents :** They enable users to read and send e-mail.
- **Message transfer agents :** They move the messages from the sender to the receiver.

Basic Functions of Email system

1. Composition 2. Transfer
3. Reporting 4. Displaying and
5. Disposition

1. Composition

- The process of creating messages and to answer them is known as composition.
- The system can also provide **assistance** with addressing and a number of header fields attached to each message.

2. Transfer

- It is the process of moving messages from the sender to the recipient.
- This includes establishment of a connection from sender to destination or some intermediate machine, transferring the message, and breaking the connection.

3. Reporting

The reporting system is designed to tell the sender about whether the message was delivered, rejected, or lost.

4. Displaying

It is the process of displaying the incoming messages so that the user can read it. For this purpose, simple conversions and formatting are required to be done.

5. Disposition

- This is concerned with what the recipient does with the received message. Disposition is the final step in e-mail system.
- **Some of the possibilities are as follows :**
  1. Throw after reading
  2. Throw before reading
  3. Save messages
  4. Forward messages
  5. Process messages in some other way.

1.15.2 Advanced Features of E-mail Systems

GQ. Explain Advanced Features of electronic mail system.

Some of the advanced features included in addition to the basic functions are as follows :

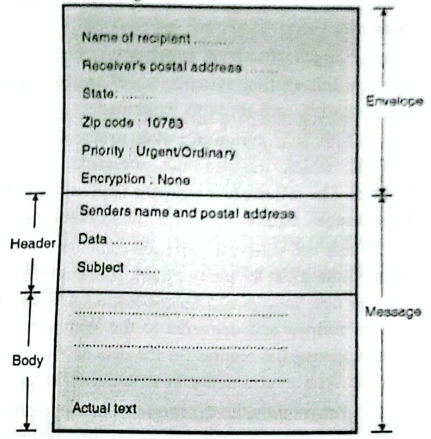
1. Forwarding an e-mail to a person away from his computer.
2. Creating and destroying mailboxes to store incoming e-mail.
3. Inspecting contents of mailbox insert and delete messages from the mailboxes.
4. Sending a message to a large group of people using the idea of mail list.
5. To provide the facility of registered e-mail.
6. Automatic notification of undelivered e-mails.
7. Carbon copies
8. High priority e-mail (setting the priority of e-mails)
9. Secret (encrypted e-mail)
10. Alternative recipient. This allows automatic forwarding of an e-mail to an alternate recipient if the main recipient is not available.

1.15.2(A) E-mail Envelope

- In the modern e-mail systems, there is a distinction made between the e-mail and its contents.
- An e-mail envelope contains the message, destination address, priority, security level etc.
- The message transport agents such as **SMTP** use this envelope for routing.

1.15.2(B) Message

- The actual message inside the envelope is made of two parts :
  1. Header 2. Body
- Header carries the control information while body contains the message contents. Envelopes and messages are shown in Fig. 1.15.1.



(G-640) **Fig. 1.15.1 : Envelope and message**

1.15.3 Message Formats

UQ. Explain Email Message formats. **(SPPU- Q. 4(b), May 15, 06 Marks)**

Let us now discuss the e-mail message formats.

1.15.3(A) RFC 822

- All the e-mail messages consist of an envelope, a few header fields, a blank line and then the message body.
- Each header field logically consists of a single line of ASCII text which consists of the field name, a colon and a field.
- Normally the user agent builds a message and passes it to the message transfer agent which uses some header fields for construction of an envelope. Table 1.15.1 shows the **principle header fields** related to the message transport. Let us discuss them one by one.

Table 1.15.1 : RFC 822 header fields related to message transport

Header name	Meaning
To :	E-mail address of primary recipients
Cc :	E-mail address of secondary recipients (Carbon copy)
Bcc :	E-mail address for blind carbon copies
From :	Originator of the message
Sender :	E-mail address of the person sending the message
Received :	Line added by each transfer agent along the route
Return – Path :	Can be used to identify the path back to the sender.

1. The To : field

This field gives the **DNS address** of the primary recipient. It is allowed to have multiple recipients.

2. The Cc : field

This field gives the addresses of any secondary recipients. Cc stands for **carbon copy**. Whatever message and attachments are sent to the primary recipient the same are sent to the secondary recipient as well.

3. The Bcc : field

- The long form of Bcc is **blind carbon copy**. This field is like Cc field, except that this is deleted from all the copies sent to the primary and secondary recipients.
- Thus, a sender can send copies to third parties without primary and secondary recipients knowing about it.

4. From: and Sender : fields

- These fields tell about who wrote the message and who actually sent the message respectively because the person who creates the message and the person who sends it can be different.
- **The From**: Field is necessary but the **Sender**: field can be omitted, if it is same as the **From** : field. These fields are required when the message cannot be delivered and is to be returned to thesender.

5. Received : field

- A line containing **Received:** is added by each message transfer agent along the way. This line carries the

agent's identity, date and time at which the message was received.

- It also contains some other information that can be used to find bugs in the routing system.

6. The Return-Path: field

The final message transfer agent adds this field and it is intended to tell how to get back to the sender. This information can be obtained from all the received : headers.

1.15.3(B) Other Header Fields

- In addition to the fields of Table 1.15.2, **RFC 822** messages may contain many other header fields. These are used by either the user agents or human recipients some of them are shown in Table 1.15.2.

Table 1.15.2 : Some fields in RFC 822 message header

Header	Meaning
Date :	The date and time of the message.
Reply-To	E-mail address to which the reply is to be sent
Message-Id :	Message identifying number
In-Reply-To :	Message-Id of the message to which this is a reply
References :	Other relevant message identifying numbers
Keywords :	Keywords chosen by user
Subject :	Summary of the message for the one-line display.

- The RFC 822 allows the users to invent new headers for their own private use, but it is essential that these headers start with the string X. For example, X-Event of the week.

1.15.3(C) Message Body

- The message body comes after the header. The users can include anything that they want to send, in the message body.
- It is possible to terminate the messages with ASCII cartoons, quotations, political statements etc.

1.16 HTTP (HYPERTEXT TRANSFER PROTOCOL)

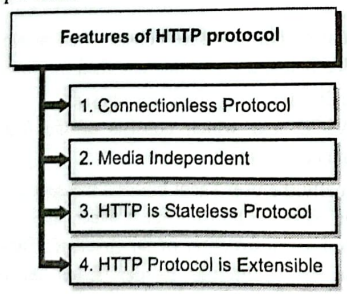
GQ. Write short note on : HTTP.

1. From 1990, the Hypertext Transfer Protocol (HTTP) is the base of data communication process in the **WWW (World Wide Web)**. HTTP protocol is used for exchanging or delivering the **hypertext over the network**.
2. It is an **application-level stateless** protocol, which is used by information systems, which are **distributed, collaborative, and contains hypermedia**.
3. The HTTP is a **communication protocol** based on TCP/IP, which transfers the information over the Internet. The information can be in the form of **HTML files, images, audios, videos, or query results**, etc.
4. HTTP can use different ports for communication, but usually, the **TCP 80 port** is used. HTTP specification denotes how the data to be requested by **Web client** will be formed and delivered to the Web server; and after receiving the request for the data at Web servers, how the Web servers will answer to the requests.
5. HTTP is also said to be the **request/response protocol**, which depends on the **client-server architecture** where the HTTP clients are: Web browsers and the HTTP Servers are Web Servers.

1.16.1 Basic Features of HTTP Protocol

GQ. Explain features of HTTP.

Following are the three features of HTTP which helps HTTP protocol to become a simple, human readable and powerful protocol :



(IF5)**Fig. 1.16.1 : Features of HTTP protocol**

1. HTTP is Connectionless Protocol

UQ. Is HTTP connection oriented or connection less protocol? **(SPPU- Q. 6(b), August 17, 04 Marks)**

(i) In HTTP protocol, after sending the HTTP request by HTTP client **(e.g. a Web browser)**, the connection between HTTP client & HTTP server is disconnected and the HTTP client waits for HTTP server's response.

(ii) Then, after receiving the HTTP client's request, the HTTP server processes the request and for sending back the response to the HTTP client, connection is re-established between the HTTP client and the HTTP server.

2. Media independent

HTTP protocol can handle any type of data over the Internet, if both the HTTP client and the HTTP server know how to handle particular type of data specified by **MIME-type**.

3. HTTP is Stateless Protocol

UQ. What is meant by statelessness of HTTP? **(SPPU- Q. 6(b), Oct. 16, 03 Marks)**

(i) The connectionless characteristic of HTTP makes the HTTP as a **stateless protocol**.

(ii) Means that, the HTTP server and HTTP client are known to each other during a current **HTTP request** only and after processing the current HTTP request both of them forget about each other.

(iii) That is why the HTTP client or the **web browser** cannot hold the information between two different HTTP requests across the web pages.

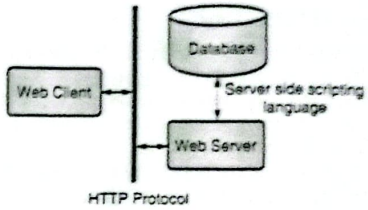
4. HTTP Protocol is Extensible

(i) In HTTP 1.0, HTTP header helps in extending the HTTP protocol, where introducing the new functionalities becomes easier.

(ii) The newer functionality is added in HTTP protocol by a simple agreement between the HTTP client and the HTTP server about new header's semantics.

1.16.1(A) Where HTTP is situated?

Fig. 1.16.2 illustrates the architecture of a web application which indicates that where the HTTP protocol is situated :



(IF5)**Fig. 1.16.2 : Architecture of web application**

1. **HTTP protocol :** HTTP is also said to be the request/response protocol, which depends on client-server architecture where the HTTP clients are Web browsers and the HTTP Servers are Web Servers.
2. **Web Client :** The HTTP client that is web browser sends a HTTP request to the HTTP server. The format of a HTTP request is as follows:
   (i) An HTTP request method,
   (ii) URI,
   (iii) HTTP Protocol version,
   (iv) A message as MIME including HTTP request modifiers, information about HTTP Client, and the body content about a TCP/IP connection.
3. **Web Server :** The HTTP server usually the Web server sends back the response. The format of a HTTP response is as follows:
   (i) A status line,
   (ii) HTTP Protocol version of message,
   (iii) A status code i.e. success or error code,
   (iv) A message as MIME including information about entity meta-information, and entity-body content.

1.16.1(B) HTTP Flow

The following steps are carried out to communicate HTTP client i.e. browser with an HTTP server i.e. an **intermediate proxy** or the server:

1. **Specify URL :** The user needs to specify URL in their browser.
2. **Open a TCP connection**

(i) Then the browser opens a TCP connection to the server using port 80.

(ii) The TCP connection is used to send one or more request / response.

(iii) The HTTP client can either open a new connection or reuse an existing one, or open number of TCP connections to the servers.

3. Send a HTTP message

(i) After connection is established, the client sends the HTTP request through this connection to the server for getting the appropriate contents. e.g. following statement is used to request a web page index.html from www.PhoenixGlobe.com.

(ii) GET www.PhoenixGlobe.com/index.html HTTP/1.0. Before the introduction of HTTP 2, HTTP messages are **human-readable**.

(iii) In HTTP 2, they are encapsulated in frames, to restrict them from direct reading, but the principle remains the same.

4. Read the response sent by the server : After getting the HTTP response sent by server through the connection, the browser renders the content on the screen according to the data given in HTTP packets and then user can read it easily.

```
HTTP/1.1 200 OK
```

Example : HTTP/1.1 200 OK indicates the response returned by sever for request (www.PhoenixGlobe.com).

5. Close or reuse the connection for further requests

(i) The **HTTP pipelining** helps for sending various HTTP requests one after another through a TCP connection regardless of whether the responses of previously sent requests are received completely or not.

(ii) It is difficult to implement HTTP Pipelining practically so in HTTP 2 this feature is supersede by the more robust concept called as **multiplexing requests** within a frame.

1.16.2 Persistent and Non-Persistent HTTP

GQ. Write short note on persistent and non-persistent HTTP.

UQ. What is persistent and non-persistent HTTP connection? **(SPPU- Q. 5(b), Aug. 15, 4 Marks)**

UQ. Explain persistent and non persistent HTTP. **(SPPU- Q. 4(b), May 16, 4 Marks)**

1. HTTP can use both nonpersistent connections and persistent connections.
2. **A nonpersistent connection** is the one that is closed after the server sends the requested object to the client. In other words, the connection is used exactly for one request and one response.
3. **With persistent connections**, the server leaves the TCP connection open after sending responses and hence the **subsequent requests** and **responses** between the same client and server can be sent.
4. The server closes the connection only when it is not used for a certain configurable amount of time. With persistent connections, the performance is improved by **20%.**

1.16.2(A) Differentiation between Non-Persistent and Persistent HTTP

GQ Compare persistent and Non persistent HTTP connections?

UQ. Differentiate between persistent and non-persistent HTTP connection. **(SPPU- Q. 6(b), Aug. 17, 4 Marks)**

Table 1.16.1 : Comparison of Non-Persistent and Persistent HTTP

Parameter	Non-Persistent HTTP	Persistent HTTP
Concept	A nonpersistent connection is the one that is **closed after the server sends** the requested object to the client.	With persistent connections, the server leaves the TCP **connection open after sending** responses and hence the subsequent requests and responses between the same client and server can be sent.
Objects	At most **one object is sent** over a TCP connection.	**Multiple objects can be sent** over a single TCP connection
RTTs	***Two RTTs** to fetch each object.*	***Fewer RTTs** and less slow start.*
HTTP version	HTTP 1.0 uses **non-persistent** HTTP.	HTTP 1.1 **uses persistent** HTTP in default mode.

1.16.3 HTTP Messages

GQ. Explain HTTP Message format.

UQ. Explain request and response message in HTTP. **(SPPU- Q. 1(b), Dec. 17, 4 Marks)**

There are two types of HTTP messages, **requests** and **responses**. Fig. 1.16.3 shows the Request/Response message format.

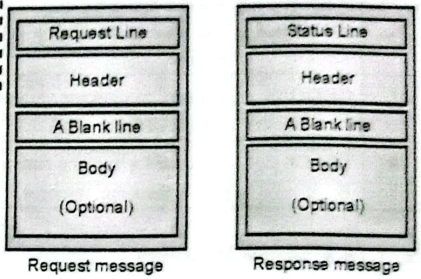


Fig. 1.16.3 : Request-Response Message Format

1. Requests

(i) An HTTP Request method: **(e.g. GET, POST) specifies the operation** to be carried out.

(ii) The URI defines **complete path of the resource** to be fetched. The HTTP protocol version being used.

(iii) **Optional headers** including **additional information** needed by the HTTP servers or a body needed by some methods like POST. Similar to responses.

(iv) The first three elements together referred as request line. Let's see more information about request line:

2. Request line

(i) The Request-Line starts with a Request-method followed by the Request-URI followed by the HTTP protocol version. All these parts are separated by a space character. **e.g. GET index.html HTTP/1.1**

(ii) Here, the Request-method is GET, the Request-URI is index.html and the HTTP protocol version is HTTP/1.1

The parts of Request-Line are explained below :

(i) The Request-method should be written in uppercase letters and it is case-sensitive.

(ii) The Table 1.16.2 lists all the Request-methods, which are supported by HTTP/1.1.

Table 1.16.2 : HTTP Request Methods

Sr. No.	Method	Description
1.	**GET**	The GET method is used for fetching the information from the server indicated by given request-URI. Requests with GET method only retrieve data from specific server and do not have any effect on the data.
2.	**HEAD**	Same as GET, means is used for retrieving data from specific server, but it transmits the **status-line** and the header section only.
3.	**POST**	Opposite of GET, the POST method is used to send data to the server. The request using POST method leads the uploading of data on the requested server. **For example**, sending the customer information, files, etc. using HTML forms.
4.	**PUT**	It is used to replace all the current target resource' contents indicated by given request URI with the uploaded content.
5.	**DELETE**	It is used to deletes all the current target resource' contents indicated by given request URI.

Sr. No.	Method	Description
6.	CONNECT	It is used to establish a path to the server indicated by given request URI for communication purpose.
7.	OPTIONS	It is used to illustrate various options of communication for the target resource indicated by given request URI.
8	TRACE	It is used for tracking purpose. It is achieved by performing a **message loop back test** through the path to the target resource.

1.16.4 Unique Identifiers

GQ. What are the different worldwide unique identifiers? Explain the components of uniform resource locators.

1.16.4(A) URL (Uniform Resource Locator)

1. A Uniform Resource Locator (URL), formally termed as a **web address**, is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it.
2. A URL is a specific type of Uniform Resource Identifier (**URI**), although many people use the two terms interchangeably.
3. URLs occur most commonly to reference web pages (http), but are also used for file transfer (**FTP**), **email** (**mailto**), **database access** (**JDBC**), and many other applications.
4. Most web browsers display the URL of a web page above the page in an address bar.
5. A typical URL could have the form **http://www.PhoenixGlobe.com/index.html**, which indicates a **protocol** (http), a **hostname** (www.PhoenixGlobe.com), and a **file name** (index.html).
6. Every HTTP URL conforms to the syntax of a generic URI. A generic URI is of the form:

   scheme: [//[user[:password]@]host[:port]][/path] [?query][#fragment]
7. The **scheme**, consisting of a sequence of characters beginning with a letter and followed by any combination of **letters, digits, plus (+), period (.), or hyphen (-).**
8. Although schemes are case-insensitive, the canonical form is lowercase and documents that specify schemes must do so with lowercase letters.
9. It is followed by a colon (:).
10. **Examples** of popular schemes include **http, https, ftp, mailto, file, data, and irc**. URI schemes should be registered with the **Internet Assigned Numbers Authority (IANA),** although non-registered schemes are also used in practice.
11. **Two slashes (//):** This is required by some schemes and not required by some others.
12. When the authority component (explained below) is absent, the path component cannot begin with two slashes.

    An authority part, comprises of :
    - An **optional authentication** section of a **username and password,** separated by a colon, followed by symbol (@)
    - A **"host"**, consisting of either a **registered name** (including but not limited to a hostname), or **an IP address.** An optional port number, separated from the hostname by a colon
13. A **path**, which contains data, usually organized in **hierarchical form** that appears as a sequence of segments separated by slashes.
14. Such a sequence may resemble or map exactly to a file system path but does not always imply a relation to one.
15. The path must begin with a **single slash (/)** if an authority part was present but must not begin with a double slash. The path is always defined, though the defined path may be empty (**zero length**), therefore no trailing slash.
16. The final part of the path may be referred to as a **'slug'.** An optional **query**, separated from the preceding part by a question mark (**?**), containing a query string of non-hierarchical data.
17. Its syntax is not well defined, but by convention is most often a sequence of attribute–value pairs separated by a delimiter. An optional **fragment**, separated from the preceding part by a **hash (#).**
18. The fragment contains a fragment identifier providing direction to a secondary resource, such as a section heading in an article identified by the remainder of the URI. When the primary resource is an **HTML document,** the fragment is often an **id attribute** of a specific element, and web browsers will scroll this element into view.
19. A web browser will usually be the rereference URL by performing an HTTP request to the specified host, by default on **port number 80.**
20. URLs using the https scheme require that requests and responses will be made over a secure connection to the website.

1.16.4(B) Request-URI

- The URI is short form of Uniform Resource Identifier. The Request-URI indicates the **requested resource**.
- Following statement specifies the most used form of an URI :

```
"*" | absoluteURI | abs_path | authority
```

- Note that the **absoluteURI** is used when the HTTP request is made to a **proxy server.** Actually, the proxy server is requested to forward the service or the request from a valid cache, and then return the response back.
- **For example:** Following statement illustrates the Request-URI, which indicates specific resource on an origin server or gateway specified in the path.

```
GET http://www.Example.com/Web/index.html HTTP/1.1
```

- Here a client wants to get back a resource (**i.e. index.html web page**) from the server (server of the site www.Example.com) directly.
- The above statement creates a TCP connection to port 80 of the host "www.Example.com" to forward the following statement :

```
GET /Web/index.html HTTP/1.1
```

1. Request Header Fields

- The fields of request-header are used to transfer additional information about the request and the client itself, to the server.
- ***Following list contains some important Request-header fields which are used on demand :***
  - Accept-Charset
  - Accept-Encoding
  - Accept-Language
  - Authorization
  - Expect
  - From
  - Host
  - If-Match
  - If-Modified-Since
  - If-None-Match
  - If-Range
  - If-Unmodified-Since
  - Max-Forwards
  - Proxy-Authorization
  - Range
  - Referer
  - TE
  - User-Agent

2. Responses

Responses includes following elements :

- The HTTP protocol version.
- A status code, i.e. **success or error code** indicating that the request has been successful, or not, respectively. If not it also tells why the error has occurred.
- A status message indicates the short description of the status code. HTTP headers similar to the headers for requests. Optional, a body containing the fetched resource.
- The first three elements together referred as a **status line.** Let's see more information about all the elements:

☞ **Message Status-Line**

- In response, the Status-Line formed by the HTTP protocol version followed by a status code (**i.e. a 3-digit number**) followed by its associated textual phrase. All the parts are separated by a space character.
- For example,

```
HTTP/1.1 200 OK
```

- Here the **HTTP/1.1** is HTTP-Version, and the Status-Code is 200 and **OK** is the Reason-Phrase. Let's see details about the parts of Status line :

(a) HTTP Version

- It indicates which HTTP protocol version is supported by the server.
- The HTTP/1.1 indicates that the server supports **HTTP 1.1 version** of the protocol.

(b) Status Code

- It is a 3-digit integer value in which only the first digit defines the **class of response** and the remaining two digits are not used for categorization purpose.
- There are **5** values for the first digit as shown in Table 1.16.3 :

Table 1.16.3 : First digit details

Sr. No.	Code	Description
1.	1xx	**Informational** It indicates that the request was received successfully, and the process is continuing.
2.	2xx	**Success** It indicates that the request was successfully received, understood, and accepted.
3.	3xx	**Redirection** It indicates that to complete the request further action should be followed.
4.	4xx	**Client Error** It indicates that the request is not successfully completed due to the presence of incorrect syntax.
5.	5xx	**Server Error** It indicates that the server is failed to fulfil a valid request.

- HTTP status-codes can be extended. It is not necessary for HTTP applications to know the meaning of all **registered. status-codes.**

Response Header Fields

- The fields of response-header are used to send additional information about the response, which cannot be put in the Status-Line.
- These response-header fields are used to give information about the server and further accessibility to the resource identified by the **Request-URI**.
- Following list contains some important **Response-header fields** which are used on demand :
  - (i) Accept-Ranges (ii) Age
  - (iii) ETag (iv) Location
  - (v) Proxy-Authenticate (vi) Retry-After
  - (vii) Server (viii) Vary
  - (ix) WWW-Authenticate

Note : The **example** of **URI** is urn: isbn: 0-476-27557-4, whereas the **example** of **URL**, is https://google.com. The **URI** can be used to find resources in HTML, XML, and other files also, whereas **URL** can only be used to locate a web page. Each **URL** can be a **URI**, whereas all **URIs** cannot always be **URLs**.

1.16.5 Stateful and Stateless Protocol

- **Network Protocols** for web browser and servers are categorized into two types:
  - Stateless Protocol, and
  - Stateful protocol.
- These two protocols are differentiated on the basis of the requirement of server or server-side software to save status or session information.

(A) Stateless Protocol

- Stateless Protocols are the type of network protocols in which Client send request to the server and server response back according to current state.
- It does not require the server to retain session information or a status about each communicating partner for multiple request.
- **Examples :** HTTP (Hypertext Transfer Protocol), UDP (User Datagram Protocol), DNS (Domain Name System) are the example of **Stateless Protocol**.

(B) Silent features of Stateless Protocols:

- Stateless Protocol **simplify** the design of Server.
- The stateless protocol requires **less resources** because system **do not need** to keep track of the multiple link communications and the session details.
- In Stateless Protocol each information packet **travel on it's own** without reference to any other packet.
- Each communication in Stateless Protocol is **discrete and unrelated** to those that precedes or follow.

(C) Stateful Protocol

In Stateful Protocol If client send a request to the server then it expects some kind of response, if it does not get any response then it resend the request. FTP (File Transfer Protocol), Telnet are the example of **Stateful Protocol**.

(D) Silent features of Stateful Protocol

- Stateful Protocols provide **better performance** to the client by keeping track of the connection information.
- Stateful Application require **Backing storage**.
- Stateful request are always **dependent** on the server-side state.
- TCP session follow stateful protocol because both systems **maintain information about the session** itself during its life.

(E) Difference between Stateless and Stateful Protocol

UQ. State Difference between Stateless and Stateful protocol. **SPPU- Q. 6(b), Oct. 17, 05 Marks**

Sr. No.	Based on	Stateless protocol	Stateful protocol
1.	**Basic**	These are the type of network protocols in which **client sends the request to the server and the server respond according to the current state.**	In stateful protocol, when a **client sends a request to the server it expects some response**, and if it does not get any response client resends the request.
2.	**Design**	It **simplifies** the server design.	It makes the server design heavy and **complex**.
3.	**Dependency**	In stateless protocol, both server and client are **independent and loosely coupled.**	While in stateful protocol, both **server and client are tightly coupled.**
4.	**Server restrictions**	In stateless protocol, server is **not restricted** to keep the server information or session details.	In stateful protocol, server is **restricted** to keep the server information or session details.
5.	**Example**	Examples of the stateless protocol are **UDP** (User Datagram Protocol), **HTTP** (Hypertext Transfer Protocol), etc.	Examples of the stateful protocol are **Telnet, FTP** (File Transfer Protocol).

Sr. No.	Based on	Stateless protocol	Stateful protocol
6.	Transaction	Transaction handling is **fast** in stateless protocol.	Transaction handling is **slower** in stateless protocol.
7.	Restoring after crash	It **works better during crash**. This is because there is no need to restore any state. It is easy to restart a failed server after crash.	It **does not work better during crash**. This is because server has to keep status information and session details.

1.17 SMTP (SIMPLE MAIL TRANSFER PROTOCOL)

GQ. Write short notes on : SMTP.

UQ. Explain in brief simple mail transfer protocol. **(SPPU- Q. 5(b), Aug. 14, Q. 3(b), May 18, Q. 3(b), Dec. 19, 05 Marks)**

1. SMTP is an acronym of **Simple Mail Transfer Protocol**. It was first proposed in 1982. It is a standard protocol, which is used for the process of sending **e-mail** over the internet with efficiency and reliability.
2. SMTP is an **application-level** protocol. SMTP is **connection-oriented** protocol. SMTP **is text-based** protocol.
3. Over the TCP/IP network, SMTP is used to exchange the messages in between e-mail servers. E-mail is delivered by having the source machine establish a TCP connection to port 25 of the destination machine.
4. Rather than sending email, one more functionality is provided by the SMTP, notification about the incoming mail.
5. In the process of email, the e-mail client sends the e-mail-to-e-mail server, which then interacts with the recipient mail server by the help of SMTP client.
6. The email addresses of sender and receiver are specified by the SMTP commands along with the message to be sent.
   - SMTP accepts incoming connections and copies messages from them into the appropriate mailboxes.
   - If a message cannot be delivered, an error report containing the first part of the undeliverable message is returned to the sender
7. There is no need of user interaction for the exchange of commands between servers
8. In case of failure in message delivery, an error report is sent to the sender because of which the SMTP is considered as a reliable protocol.

1.17.1 SMTP Commands

GQ. List and Describe SMTP commands.

Table 1.17.1 : SMTP commands

Sr. No.	Command	Description
1.	HELLO	This command is used to initiate the conversation of SMTP.
2.	EHELLO	This is substitute command HELLO. E specifies that the sender server like to use extended SMTP protocol.
3.	MAIL FROM	This specifies the address of sender.
4.	RCPT TO	It specifies the address of recipient of the mail.
5.	SIZE	This command specifies size of attached message in bytes.

Sr. No.	Command	Description
6.	DATA	IT specifies the body (actual content) of the message.
7.	QUIT	This command terminates the SMTP connection.
8.	VERFY	This command is used to check the validity of given username.
9.	EXPN	It is same as VERFY, except it displays the list of all the user names when it used with a distribution list.

Response Code

- **Response** are sent from the server to the client. A response is a **three-digit code** that may be followed by additional textual information. Following are few examples of code and its description
- **220 -** SMTP Service ready
- **250 -** Your message was delivered to the recipient server.
- **421 -** The service is not available and the connection will be closed.
- **450 -** The requested command failed because the user's mailbox was unavailable
- **500 -** The server could not recognize the command due to a syntax error.

1.17.2 SMTP Header Format

UQ. Describe SMTP header format. **(SPPU- Q. 4(b), May 15, 06 Marks)**

Electronic Mail (e-mail) is one of the most widely used services of the Internet. This service allows an Internet user to send a **message in a formatted manner (mail)** to other Internet users in any part of the world. Message in the mail not only contain text, but it also contains images, audio and videos data. The person who is sending mail is called **sender** and person who receives mail is called the **recipient**. It is just like postal mail service.

Format of E-mail

An e-mail consists of three parts that are as follows :

1. Envelope **2.** Header **3.** Body

These are explained as following below.

1. Envelope

The envelope part encapsulates the message. It contains all information that is required for sending any e-mail such as destination address, priority and security level. The envelope is used by MTAs for routing message.

2. Header

The header consists of a series of lines. Each header field consists of a single line of ASCII text specifying field name, colon and value. The main header fields related to message transport are :

1. **To :** It specifies the DNS address of the primary recipient(s).
2. **Cc :** It refers to carbon copy. It specifies address of secondary recipient(s).
3. **BCC :** It refers to blind carbon copy. It is very similar to Cc. The only difference between Cc and Bcc is that it allow user to send copy to the third party without primary and secondary recipient knowing about this.
4. **From :** It specifies name of person who wrote message.
5. **Sender :** It specifies e-mail address of person who has sent message.
6. **Received :** It refers to identity of sender's, data and also time message was received. It also contains the information which is used to find bugs in routing system.
7. **Return-Path :** It is added by the message transfer agent. This part is used to specify how to get back to the sender.

3. Body

The body of a message contains text that is the actual content/message that needs to be sent, such as "Employees who are eligible for the new health care program should contact their supervisors by next Friday if they want to switch." The message body also may include signatures or automatically generated text that is inserted by the sender's email system.

The above-discussed field is represented in tabular form as follows :

Main From	ErManojkavedia@gmail.com	Envelope
RCPT To	profmanojkavedia@gmail.com	
From	ErManojkavedia@gmail.com	Header
To	profmanojkavedia@gmail.com	
Date	Mon,29 Feb 2020 10:13:17 EST	
subject	Birthday Wishes	
Welcome to my new Venture KaizenFutureTech. We Deal in Embedded System, IoT, Project and Training Program related to IoT, Embedded, and Econtent development. Thanks Manoj Kavedia		Body

Fig. 1.17.1 : SMTP / Email format

Header	Meaning
To:	E-mail address of primary receipt(s).
Cc:	E-mail address of secondary receipt(s).
Bcc:	E-mail address for blind carbon copies.
From:	Person or people who have created a message.
Sender:	E-mail address of the actual sender.
Received:	It is used to specify how to get back to the sender.
Return-Path	It can be used to identify a path back to the sender.

In addition to above-discussed fields, the header may also contain a variety of other fields which are as follows :

Header	Meaning
Date:	Date and time when the message was sent.
Reply-To:	It contains e-mail address to which replies should be sent.
Message-Id:	It refers to the unique number for referencing this message later.
In-Reply-To:	Message-Id of a message to which this is as a reply.
References:	It contains other relevant message-ids.
Keywords:	User-chosen keywords.
Subject:	It contains short summary of message for one-line display.

1.18 POP PROTOCOL (POST OFFICE PROTOCOL)

GQ. How does electronic mail system work? What is the role of SMTP and POP-3 server in E-mail system?

GQ. State and explain Post office Protocol?

UQ. Write short note on POP3.

(SPPU- Q. 4(a), May 19, 06 Marks)

UQ. Explain the role of SMTP and POP3 protocols in transfer of e-mail message.

(SPPU- Q. 5(a), Oct. 16, 05 Marks)

The POP protocol stands for **Post Office Protocol**. As we know, that SMTP is used as a message transfer agent. When the message is sent, then SMPT is used to deliver the message from the client to the server and then to the recipient server. But the message is sent from the recipient server to the actual server with the help of the Message Access Agent.

The Message Access Agent contains two types of protocols, i.e.,

- POP3 and IMAP.

How mail is transmitted ?

- Suppose sender wants to send the mail to receiver. First mail is transmitted to the sender's mail server. Then, the mail is transmitted from the sender's mail server to the receiver's mail server over the internet.
- On receiving the mail at the receiver's mail server, the mail is then sent to the user. The whole process is done with the help of Email protocols.
- The transmission of mail from the sender to the sender's mail server and then to the receiver's mail server is done with the help of the **SMTP protocol**.

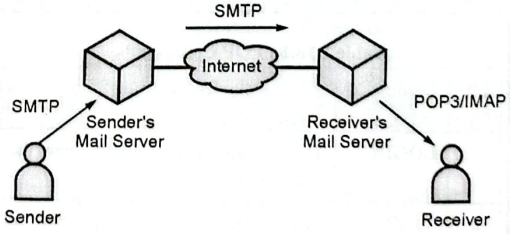


Fig. 1.18.1 : How Mail is Transmitted

- At the receiver's mail server, the POP or IMAP protocol takes the data and transmits to the actual user. Since SMTP is a push protocol so it pushes the message from the client to the server. As we can observe in the above figure that SMTP pushes the message from the client to the recipient's mail server.
- The third stage of email communication requires a pull protocol, and POP is a pull protocol. When the mail is transmitted from the recipient mail server to the client, which means that the client is pulling the mail from the server.

1.18.1 POP3 (Post Office Protocol)

- POP is an acronym of Post Office Protocol. POP is basically designed to support single client. Now a day several versions of POP are available but the POP 3 is considered as the current standard.
- POP is an **application layer internet standard** protocol. POP requires very less time of internet usage as it supports offline access to the messages.
- Search facility is not provided by the POP. It is necessary to download the messages to access them. In POP, only one mailbox can be created on the server.
- POP cannot access non-mail data. Commands regarding the POP are usually abbreviated into codes of three to four letters. **Eg. LIST.**

1.18.2 History of POP3 Protocol

- The first version of post office protocol was first introduced in 1984 as RFC 918 by the internet engineering task force. The developers developed a simple and effective email protocol known as the POP3 protocol, which is used for retrieving the emails from the server. This provides the facility for accessing the mails offline rather than accessing the mailbox offline.
- In 1985, the post office protocol version 2 was introduced in RFC 937, but it was replaced with the post office protocol version 3 in 1988 with the publication of **RFC 1081**. Then, POP3 was revised for the next 10 years before it was published. Once it was refined completely, it was published on 1996.
- Although the POP3 protocol has undergone various enhancements, the developers maintained a basic principle that it follows a three-stage process at the time of mail retrieval between the client and the server. They tried to make this protocol very simple, and this simplicity makes this protocol very popular today.

1.18.3 Working of the POP3 Protocol

UQ. Describe working of POP3 protocol.
(SPPU- Q. 7(a), Dec. 14, 08 Marks)

- To establish the connection between the POP3 server and the POP3 client, the POP3 server asks for the username to the POP3 client. If the username is found in the POP3 server, then it sends the **OK message**.
- It then asks for the **password** from the POP3 client, then the POP3 client sends the password to the POP3 server. If the password is matched, then the POP3 server sends the OK message, and the connection gets established.

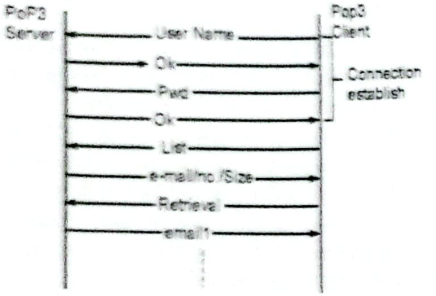


Fig. 1.18.2 : POP Version3

- After the establishment of a connection, the client can see the list of mails on the POP3 mail server. In the list of mails, the user will get the email numbers and sizes from the server. Out of this list, the user can start the retrieval of mail.
- Once the client retrieves all the emails from the server, all the emails from the server are deleted. Therefore, we can say that the emails are restricted to a particular machine, so it would not be possible to access the same mails on another machine.
- This situation can be overcome by configuring the email settings to leave a copy of mail on the mail server.

1.18.4 Advantages and Disadvantages of POP3 Protocol

Advantages

1. It allows the users to read the email **offline**. It requires an internet connection only at the time of downloading emails from the server. Once the mails are downloaded from the server, then all the downloaded mails reside on our PC or hard disk of our computer, which can be accessed without the internet. Therefore, we can say that the POP3 protocol does not require permanent internet connectivity.
2. It provides easy and fast access to the emails, as they are already stored on our PC.
3. There is no limit on the size of the email, which we receive or send.
4. It requires less server storage space as all the mails are stored on the local machine.
5. There is maximum size on the mailbox, but it is limited by the size of the hard disk.
6. It is a simple protocol so it is one of the most popular protocols used today.
7. It is easy to configure and use.

Disadvantages

1. If the emails are downloaded from the server, then all the mails are deleted from the server by default. So, mails cannot be accessed from other machines unless they are configured to leave a copy of the mail on the server.
2. Transferring the mail folder from the local machine to another machine can be difficult.
3. Since all the attachments are stored on your local machine, there is a high risk of a virus attack if the virus scanner does not scan them. The virus attack can harm the computer.
4. The email folder, which is downloaded from the mail server, can also become corrupted.
5. The mails are stored on the local machine, so anyone who sits on your machine can access the email folder.

1.18.5 POP Commands

GQ. Explain POP commands.

Table 1.18.1 : POP Commands

Sr. No.	Command	Description
1.	LOGIN	This command is used to open the connection.
2.	STAT	This command shows number of current messages in the mailbox.
3.	LIST	This command creates the summary of messages.

Sr. No.	Command	Description
4.	RETR	This command selects a mailbox in order to access the messages.
5.	DELE	This command deletes a message.
6.	RSET	This command reset the session to its initial state.
7.	QUIT	This command helps to log off the session.

1.19 IMAP (INTERNET MESSAGE ACCESS PROTOCOL)

UQ. Write short note on IMAP.
(SPPU- Q. 4(a), May 19, 06 Marks)

- IMAP stands for **Internet Message Access Protocol**. It is an application layer protocol, which is used to receive the emails from the mail server. It is the most commonly used protocols like **POP3** for retrieving the emails.
- It also follows the **client/server model**. On one side, we have an IMAP client, which is a process running on a computer.
- On the other side, we have an IMAP server, which is also a process running on another computer. Both computers are connected through a network.

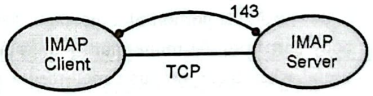


Fig. 1.19.1 : IMAP

- The IMAP protocol resides on the TCP/IP transport layer, which means that it implicitly uses the reliability of the protocol. Once the TCP connection is established between the IMAP client and IMAP server, the IMAP server listens to the port 143 by default, but this port number can also be changed.
- **By default, there are two ports used by IMAP:**
  - **Port 143 :** It is a non-encrypted IMAP port.
  - **Port 993 :** This port is used when IMAP client wants to connect through IMAP securely.

Why is IMAP used instead of POP3 protocol?

- POP3 is becoming the most popular protocol for accessing the TCP/IP mailboxes. It implements the offline mail access model, which means that the mails are retrieved from the mail server on the local machine, and then deleted from the mail server. Nowadays, millions of users use the POP3 protocol to access the incoming mails.
- Due to the offline mail access model, it cannot be used as much. The online model we would prefer in the ideal world. In the online model, we need to be connected to the internet always. The biggest problem with the offline access using POP3 is that; the mails are permanently removed from the server, so multiple computers cannot access the mails. The solution to this problem is to store the mails at the remote server rather than on the local server. The POP3 also faces another issue, i.e., data security and safety.
- The solution to this problem is to use the disconnected access model, which provides the benefits of both online and offline access. In the disconnected access model, the user can retrieve the mail for local use as in the POP3 protocol, and the user does not need to be connected to the internet continuously.
- However, the changes made to the mailboxes are synchronized between the client and the server. The mail remains on the server so different applications in the future can access it. When developers recognized these benefits, they attempted to implement the disconnected access model.
- This is implemented by using the POP3 commands that provide the option to leave the mails on the server. This works, but only to a limited extent, for example, keeping track of which messages are new or old become an issue when both are retrieved and left on the server. So, the POP3 lacks some features which are required for the proper disconnected access model.
- In the mid-1980s, the development began at Stanford University on a new protocol that would provide a more capable way of accessing the user mailboxes. The result was the development of the interactive mail access protocol, which was later renamed as **Internet Message Access Protocol**.

When to use POP3 and IMAP

UQ. When POP3 and IMAP is used in electronic mail. **(SPPU- Q. 8(a), Aug. 14, 05 Marks)**

- The Internet Message Access Protocol (IMAP) is a mail protocol used for accessing email on a remote web server from a local client. IMAP and POP3 are the two most commonly used Internet mail protocols for **retrieving emails**. Both protocols are supported by all modern email clients and web servers.
- While the POP3 protocol assumes that your email is being accessed only from one application, IMAP allows simultaneous access by multiple clients.
- This is why IMAP is more suitable for you if you're going to access your email from different locations or if your messages are managed by multiple users.

1.19.1 IMAP History and Standards

- The first version of IMAP was formally documented as an internet standard was IMAP version 2, and in RFC 1064, and was published in July 1988.
- It was updated in RFC 1176, August 1990, retaining the same version. So, they created a new document of version 3 known as IMAP3. In RFC 1203, which was published in February 1991. However, IMAP3 was never accepted by the marketplace, so people kept using IMAP2.
- The extension to the protocol was later created called IMAP is, which added support for Multipurpose Internet Mail Extensions (MIME) to IMAP. This was a very important development due to the usefulness of MIME. Despite this, IMAPbis was never published as an RFC. This may be due to the problems associated with the IMAP3.
- In December 1994, IMAP version 4, i.e., IMAP4 was published in two RFCs, i.e., RFC 1730 describing the main protocol and RFC 1731 describing the authentication mechanism for IMAP 4. IMAP 4 is the current version of IMAP, which is widely used today.
- It continues to be refined, and its latest version is actually known as IMAP4rev1 and is defined in RFC 2060. It is most recently updated in RFC 3501.

1.19.2 IMAP Features

SQ. List and Explain features of IMAP.

IMAP was designed for a specific purpose that provides a more flexible way of how the user accesses the mailbox. It can operate in any of the three modes, i.e., **online, offline, and disconnected mode.** Out of these, offline and disconnected modes are of interest to most users of the protocol.

The following are the features of an IMAP protocol :

- **Access and retrieve mail from remote server:** The user can access the mail from the remote server while retaining the mails in the remote server.
- **Set message flags:** The message flag is set so that the user can keep track of which message he has already seen.
- **Manage multiple mailboxes:** The user can manage multiple mailboxes and transfer messages from one mailbox to another. The user can organize them into various categories for those who are working on various projects.
- **Determine information prior to downloading:** It decides whether to retrieve or not before downloading the mail from the mail server.
- **Downloads a portion of a message:** It allows you to download the portion of a message, such as one body part from the mime-multi part. This can be useful when there are large multimedia files in a **short-text element** of a message.
- **Organize mails on the server:** In case of POP3, the user is not allowed to manage the mails on the server. On the other hand, the users can organize the mails on the server according to their requirements like they can create, delete or rename the mailbox on the server.
- **Search:** Users can search for the contents of the emails.
- **Check email-header:** Users can also check the email-header prior to downloading.
- **Create hierarchy:** Users can also create the folders to organize the mails in a hierarchy.

1.19.3 Working of IMAP

UQ. Explain working of IMAP. **(SPPU- Q. 1(a), May 14, 05 Marks)**

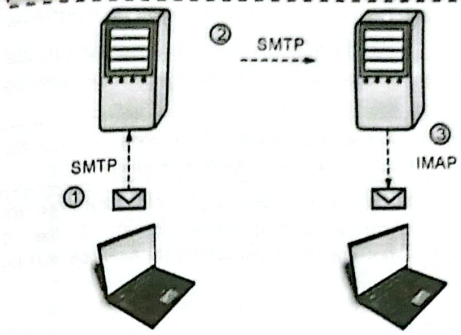


Fig. 1.19.2 : Working of IMAP

1. The IMAP is a **client-server protocol** like POP3 and most other TCP/IP application protocols. The IMAP4 protocol functions only when the **IMAP4** must reside on the server where the user mailboxes are located.
2. In client-server the POP3 does not necessarily require the same **physical server** that provides the SMTP services. Therefore, in the case of the IMAP protocol, the mailbox must be accessible to both SMTP for incoming mails and IMAP for retrieval and modifications.
3. The IMAP uses the **Transmission Control Protocol (TCP)** for communication to ensure the delivery of data and received in the order.
4. The IMAP4 listens on a well-known port, **i.e., port number 143**, for an incoming connection request from the IMAP4 client.

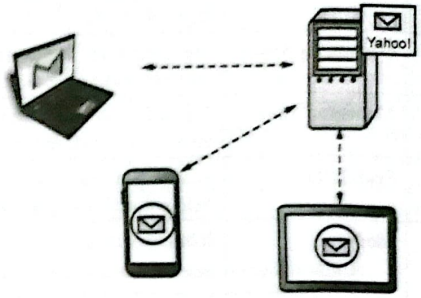


Fig. 1.19.3 : Working of IMAP

- The IMAP protocol synchronizes all the devices with the main server. Let's suppose we have three devices desktop, mobile, and laptop as shown in the Fig. 1.19.3.
- If all these devices are accessing the same mailbox, then it will be synchronized with all the devices. Here, synchronization means that when mail is opened by one device, then it will be marked as opened in all the other devices, if we delete the mail, then the mail will also be deleted from all the other devices.
- Therefore, we have synchronization between all the devices. In IMAP, we can see all the folders like spam, inbox, sent, etc. We can also create our own folder known as a custom folder that will be visible in all the other devices.

1.19.4 Difference Between POP3 and IMAP

UQ. State difference between POP3 and IMAP. **(SPPU- Q. 7(a), May 15, 08 Marks)**

Table 1.19.1 : Comparison of POP3 and IMAP

Sr. No.	Parameters	Post Office Protocol (POP3)	Internet Message Access Protocol (IMAP)
1.	**Meaning**	POP is a simple protocol that only allows downloading messages from your Inbox to your local computer.	IMAP is much more advanced and allows you the user to see all the folders on the mail server.
2.	**Port No**	The POP server listens on port 110, and the POP with SSL secure(POP3DS) server listens on port 995	The IMAP server listens on port 143, and the IMAP with SSL secure(IMAPDS) server listens on port 993.
3.	**No of devices**	In POP3 the mail can only be accessed from a single device at a time.	Messages can be accessed across multiple devices

Sr. No.	Parameters	Post Office Protocol (POP3)	Internet Message Access Protocol (IMAP)
4.	Read Mail	To read the mail it has to be downloaded on the local system.	The mail content can be read partially before downloading.
5.	Organizing Mails	The user can not organize mails in the mailbox of the mail server.	The user can organize the emails directly on the mail server.
6.	Managing Mails	The user can not create, delete or rename email on the mail server.	The user can create, delete or rename email on the mail server.
7.	Search the content	A user can not search the content of mail before downloading to the local system.	A user can search the content of mail for specific string before downloading.
8.	Modes	It has two modes : delete mode and keep mode. In delete mode, the mail is deleted from mail box after retrieval. In keep mode, the mail remains in the mail box after retrieval.	Multiple redundant copies of the message are kept at the mail server, in case of loss of message of a local server, the mail can still be retrieved

1.20 MIME (MULTIPURPOSE INTERNET MAIL EXTENSION)

GQ. Write a short notes on : MIME.

UQ. What is MIME ? Discuss its role in SMTP. (SPPU- Q. 3(b), Dec. 16, May 19, 04 Marks)

UQ. What is MIME ? Explain the need of MIME with suitable example. (SPPU- Q. 5(b), Aug. 17, 04 Marks)

MIME stands for Multipurpose Internet Mail Extension. These types help to e-mails to include information other than plain text.

- **MIME media types indicate the following things :**
  - The way of combining various parts of a message like text and attachments.
  - The way of specifying each part of the message.
  - The way of encoding different items for transmission so that the basic software which is designed to handle only with ASCII text can process the message.
- Nowadays, MIME types are not just used in the process of handling e-mail; they are now adopted by Web servers as a way to tell Web browsers the type of content being sent to them so that they should be able to cop with message in perfect manner.
- **MIME content type consists of two parts :**
  - A main type
  - A sub-type
- The main type is separated from the subtype by a forward slash character. For example, text/html for HTML.
  - text
  - message
  - audio
  - video
  - image
  - multipart
  - application
  - model
- **For example, the text main type can have types of plain text files, like :**
  - text/plain for plain text files,
  - text/html for HTML files,
  - text/rtf for text files using rich text formatting,
- The Internet Assigned Numbers Authority (IANA) assigns and lists the MIME types. When specifying the MIME type of a content-type field you can also indicate the character set for the text being used. If you do not specify a character set, the default is US-ASCII.

  For example - content-type:text/plain; charset=iso-8859-1.

Why do we need MIME?

Limitations of Simple Mail Transfer Protocol (SMTP)

1. SMTP has a very simple structure
2. It's simplicity however comes with a price as it only send messages in NVT 7-bit ASCII format.
3. It cannot be used for languages that do not support 7-bit ASCII format such as- French, German, Russian, Chinese and Japanese, etc. so it cannot be transmitted using SMTP. So, in order *to make SMTP more broad we use MIME.*
4. It cannot be used to send binary files or video or audio data.

Purpose and Functionality of MIME

- Growing demand for Email Message, as people also want to express in terms of Multimedia. So, MIME another email application is introduced, as it is not restricted to textual data.
- MIME *transforms non-ASCII data* at sender side to **NVT 7-bit data** and delivers it to the client SMTP. The message at receiver side is transferred back to the original data. As well as we can send video and audio data using MIME as it transfers them also in 7-bit ASCII data.

Features of MIME

1. It is able to send multiple attachments with a single message.
2. Unlimited message length.
3. Binary attachments (executable, images, audio, or video files) which may be divided if needed.
4. MIME provided support for varying content types and multi-part messages.

Working of MIME

- Suppose a user wants to send an email through user agent and it is in a non-ASCII format so there is a MIME protocol which converts it into **7-bit NVT ASCII** format.
- Message is transferred through e-mail system to the other side in 7-bit format now MIME protocol again converts it back into **non-ASCII code** and now the user agent of receiver side reads it and then information is finally read by the receiver.
- MIME header is inserted at the beginning of any e-mail transfer.

MIME with SMTP and POP

- SMTP transfers the mail being a message transfer agent from sender's side to the mailbox of receiver side and stores it and MIME header is added to the original header and provides additional information.
- While POP being the message access agent organizes the mails from the mail server to the receivers computer. POP allows user agent to connect with the message transfer agent.

1.20.1 MIME Headers

GQ. List and Explain MIME Headers.

- Multipurpose Internet Mail Extensions (MIME) headers appear at the beginning of a MIME message as well as within the separate body parts. Some MIME headers can be used both as message headers and in MIME body parts. Some additional headers are defined for use only in body parts.
- The following headers are defined in MIME :

1. MIME-Version
2. Content-Type
3. Content-Disposition
4. Content-Transfer-Encoding

1. MIME-Version : The presence of this header indicates the message is MIME-formatted. The value is typically **"1.0"** so this header appears as

```
MIME-Version: 1.0
```

2. Content-Type : This header indicates the media type of the message content, consisting of a ***type*** **and** ***subtype***, for example

```
Content-Type: text/plain
```

- Through the use of the multipart type, MIME allows mail messages to have parts arranged in a tree structure where the leaf nodes are any non-multipart content type and the non-leaf nodes are any of a variety of multipart types. This mechanism supports:
- simple text messages using text/plain (the default value for "Content-Type: ")
- **Text plus attachments** (multipart/mixed with a text/plain part and other non-text parts). A MIME message including an attached file generally indicates the file's original name with the "**Content-disposition**:" header, so the type of file is indicated both by the MIME content-type and the (usually OS-specific) filename extension reply with original attached (multipart/mixed with a text/plain part and the original message as a message/rfc822 part)
- Alternative content, such as a message sent in both plain text and another format such as HTML (multipart/alternative with the same content in text/plain and text/html forms).

- Image, audio, video and application (for example, image/jpeg, audio/mp3, video/mp4, and application/msword and so on) many other message contracts.

3. Content-Disposition

- The original MIME specifications only described the structure of mail messages. They did not address the issue of presentation styles.
- The content-disposition header field was added in **RFC 2183** to specify the presentation style. A MIME part can have:
  - An online **content-disposition**, which means that it should be automatically displayed when the message is displayed, or
  - An attachment content-disposition, in which case it is not displayed automatically and requires some form of action from the user to open it.
- In addition to the presentation style, the content-disposition header also provides fields for specifying the name of the file, the creation date and modification date, which can be used by the reader's mail user agent to store the attachment.
- The following example is taken from RFC 2183, where the header is defined

```
Content-Disposition: attachment; filename=genome.jpeg;
modification-date="Wed, 12 Feb 1997 16:29:51 -0500";
```

4. Content-Transfer-Encoding[edit]

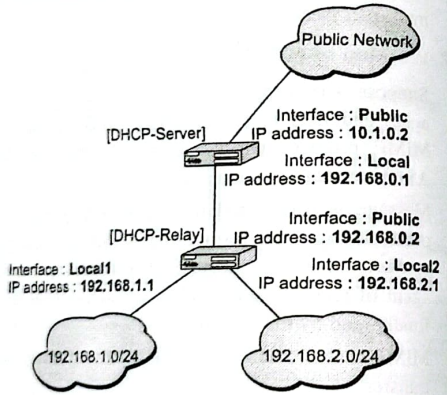
- In June 1992, MIME (RFC 1341, since made obsolete by RFC 2045) defined a set of methods for representing binary data in formats other than ASCII text format. The content-transfer-encoding MIME header has 2-sided significance:
- It indicates whether or not a binary-to-text encoding scheme has been used on top of the original encoding as specified within the Content-Type header:
- If such a **binary-to-text encoding** method has been used, it states which one. If not, it provides a descriptive label for the format of content, with respect to the presence of 8-bit or binary content.

1.21 DHCP (DYNAMIC HOST CONFIGURATION PROTOCOL)

UQ. Is static and Dynamic IP address allocation supported by DHCP protocol? Explain the state transition diagram of DHCP client and explain in brief. **(SPPU- Q. 1(b), Aug. 17, 04 Marks)**

UQ. What is DHCP ? **(SPPU- Q. 1(a), May 18, Q. 2(b), Dec. 19, 2 Marks)**

- Dynamic Host Configuration Protocol (DHCP) is defined as a network protocol that initiates or enables a server to **automatically allocate an IP address** to a respective computer from a specific defined range of numbers **(i.e., defined as scope)** which is configured for a required network.
- DHCP will assign new IP addresses in each location when devices are moved from place to place, which means network administrators do not have to manually initially configure each device with a valid IP address or reconfigure the device with a new IP address if it moves to a new location on the network.
- Versions of DHCP are available for use in **Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6).**



(1F10)**Fig. 1.21.1 : DHCP**

1.21.1 Need of DHCP

GQ. Why we need DHCP ? Explain in detail.

1. It is used to **avoid manual entry** of IP address and configuration information for all the computers/ network devices individually.
2. To prevent two computers/ network devices from having the same IP address (accidentally - due to **manual configuration errors**).
3. To provide a **central mechanism** to keep track of all the assigned IP addresses in the network.
4. To automatically **reclaim unused IP addresses** and re-assign them to other devices in the network.
5. To make **central configuration/ topology** changes in the network easier.
6. To dynamically allocate IP addresses to portable clients / wireless devices that keeps moving from one place to another in the network.

Advantages and Disadvantages of DHCP

UQ. What is its advantage of DHCP ? **(SPPU- Q. 1(a), May 18, 2 Marks, Q. 2(b), Dec. 19, 6 Marks)**

Advantages

- DHCP is easy to implement and does automatic assignment of IP addresses to requesting clients. Hence manual configuration time of IP addresses can be reduced.
- The implementation does not require any additional costs.
- Duplicate or invalid assignment of IP addresses are prevented. Hence there is no chance of conflicts in IP addresses.
- It simplifies administration of the network.
- It supports multiple scopes e.g. multicast scope, super scope etc.
- It has great benefit to mobile users as valid configuration parameters are automatically obtained from the new network.

Disadvantages

- DHCP server can be single point of failure in networks having only one configured DHCP server.
- DHCP packets can not travel across router, Hence relay agent is necessary to have DHCP server handle all leases on both network segments. Relay agents receive broadcast DHCP packets and forward them as unicast packets to DHCP server. Here relay agent must be configured with IP address of the DHCP server.
- Security: As DHCP server has no secure mechanism for authentication of the client, it can gain unauthorized access to IP addresses by presenting credentials such as client identifiers which belong to other DHCP clients.
- The machine name does not change when new IP address is assigned. Client is not able to access the network in the absence of the DHCP server.

1.21.2 DHCP Operation

UQ. Explain DHCP Operation in detail. **(SPPU- Q. 2(b), Oct. 15, 06 Marks)**

1. DHCP runs at the **application layer** of the Transmission Control Protocol/IP (TCP/IP) protocol stack to dynamically assign IP addresses to DHCP clients and to allocate TCP/IP configuration information to DHCP clients. This includes subnet mask information, default gateway IP addresses and domain name system (DNS) addresses.
2. DHCP is a client-server protocol in which servers manage a pool of unique IP addresses, as well as information about client configuration parameters, and assign addresses out of those address pools. DHCP - enabled clients send a request to the DHCP server whenever they connect to a network.
3. Clients configured with DHCP broadcast a request to the DHCP server and request network configuration information for the local network to which they are attached. A client typically broadcasts a query for this information immediately after booting up.
4. The DHCP server responds to the client request by providing IP configuration information previously specified by a network administrator. This includes a specific IP address as well as for the time period, also called **a lease**, for which the allocation is valid.
5. When refreshing an assignment, a DHCP client requests the same parameters, but the DHCP server may assign a new IP address based on policies set by administrators.
6. A DHCP server manages a record of all the IP addresses it allocates to network nodes. If a node is relocated in the network, the server identifies it using its Media Access Control (MAC) address, which prevents accidentally configuring multiple devices with the same IP address.

(New Syllabus w.e.f academic year 21-22) (P6-55) Tech-Neo Publications ...A SACHIN SHAH Venture

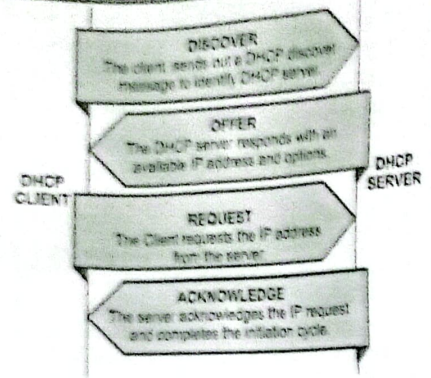


Fig. 1.21.2 : DHCP Handshake

7. DHCP is not a routable protocol, nor is it a secure one. DHCP is limited to a specific **local area network (LAN)**, which means a single DHCP server per LAN is adequate, or two servers for use in case of a failover.
8. Larger networks may have a **wide area network (WAN)** containing multiple individual locations. Depending on the connections between these points and the number of clients in each location, multiple DHCP servers can be set up to handle the distribution of addresses.
9. If network administrators want a DHCP server to provide addressing to multiple subnets on a given network, they must configure DHCP relay services located on interconnecting routers that DHCP requests have to cross.
10. These agents relay messages between DHCP clients and servers located on different subnets. DHCP lacks any built-in mechanism that would allow clients and servers to authenticate each other.
11. Both are vulnerable to deception (one computer pretending to be another) and to attack, where rogue clients can exhaust a DHCP server's IP address pool.

1.21.2(A) Features of IP Address

- **IP Address Range :** Certain IP addresses (and IP address ranges) can be excluded from being allocated to clients by the **DHCP Server**. These might be static IP addresses that are assigned to the servers, for example. It is also possible to assign the same IP addresses to certain network devices repeatedly, by the DHCP Server itself.
- The range of IP addresses that can be allocated by the DHCP Server needs to be specified by the administrator.
- **Redundancy :** When the lease period expires, the client will normally contact the DHCP Server to renew the lease. If the server is up, it will be renewed.
- However, if the server is not functioning, the client will broadcast the renew request to the network hoping that some other DHCP server in the network can renew the lease, if it is configured to do so. Therefore, some level of redundancy is built into DHCP.
- **Security :** DHCP does not provide much of security, as there is no authentication process for either the DHCP client or the DHCP server. Therefore, it is possible for rogue servers to claim as genuine DHCP servers and rogue clients to overwhelm the DHCP server with too many false IP address requests.

1.21.3 DHCP Message Format

GQ. Draw and explain function of each field of DHCP message format?
UQ. Explain various messages used in DHCP. **(SPPU- Q. 1(a), May 18, 4 Marks, Q. 2(b), Dec. 19, 6 Marks)**

- All Dynamic Host Configuration Protocol (DHCP) messages include a **FIXED format** section and a **VARIABLE format** section. The fixed format section consists of several fields that are same in every Dynamic Host Configuration Protocol (**DHCP**) message.
- The variable format section in the Dynamic Host Configuration Protocol (DHCP) contains "OPTIONS", which carry additional configuration parameters.

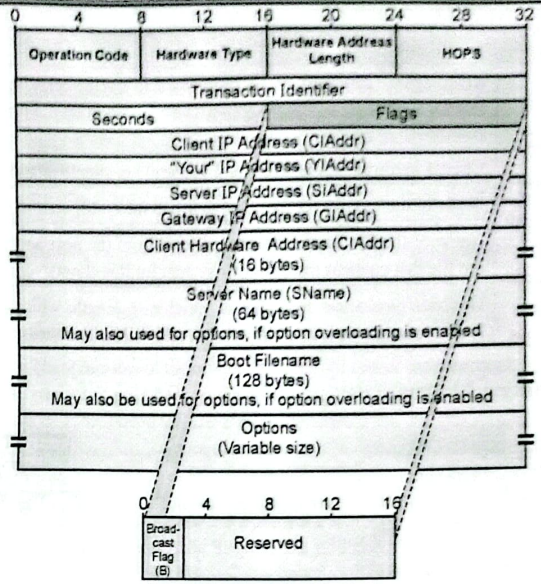


Fig. 1.21.3 : DHCP Message format

Table 1.21.1 : DHCP message field description

DHCP Message Field	Description
Operation Code (op)	Specifies the type of the Dynamic Host Configuration Protocol (DHCP) message. Set to 1 in messages sent by a client (requests) and 2 in messages sent by a server (response).
Hardware Type (Htype)	Specifies the network LAN architecture. For example, the ethernet type is specified when htype is set to 1, IEEE802 Network-6, HDLC-17, Fibre Optic-18, ATm-19, Serial line-20
Hardware Address Length (HLEN)	Layer 2 (Data-link layer) address length (MAC address) (in bytes); defines the length of hardware address in the chaddr field. For Ethernet (Most widely used LAN Standard), this value is 6.
Hops	Number of relay agents that have forwarded this message.
Transaction identifier (xid)	Used by clients to match responses from servers with previously transmitted requests.
Seconds (secs)	Elapsed time (in seconds) since the client began the Dynamic Host Configuration Protocol (DHCP) process.
Flags	Flags field is called the broadcast bit, can be set to 1 to indicate that messages to the client must be broadcast
ciaddr	Client's IP address; set by the client when the client has confirmed that its IP address is valid.
yiaddr	Client's IP address; set by the server to inform the client of the client's IP address.
siaddr	IP address of the next server for the client to use in the configuration process (for example, the server to contact for TFTP download of an operating system kernel).

DHCP Message Field	Description
giaddr	Relay agent (gateway) IP address; filled in by the relay agent with the address of the interface through which Dynamic Host Configuration Protocol (DHCP) message was received.
chaddr	Client's hardware address (Layer 2 address) i.e. clients IP address.
sname	Name of the next server for client to use in the configuration process.
file	Name of the file for the client to request from the next server (for example the name of the file that contains the operating system for this client).
options	Optional parameters field that is variable in length, which includes the message type, lease, domain name server IP address, and WINS IP address.

1.21.4 DHCP Transition Diagram

UQ. Explain address assignment in DHCP in detail with transition diagram. (SPPU- Q. 2(b), Oct. 16, 06 Marks)
UQ. Explain working of DHCP with transition diagram. (SPPU- Q. 4(a), Dec. 15, 06 Marks)
UQ. Explain the state transition diagram of DHCP client and explain in brief. (SPPU- Q. 1(b), Aug. 17, 04 Marks)

- DHCP is Dynamic Host Configuration Protocol for assigning IP addresses to devices on a network, a device can have different IP address every time it connects to the network.
- The DHCP has been devised to provide static and dynamic address allocation.
- To provide dynamic address allocation, the DHCP client acts as a state machine that performs transitions from one state to another depending on the messages it receives or sends. Figure shows the transition diagram with main states.

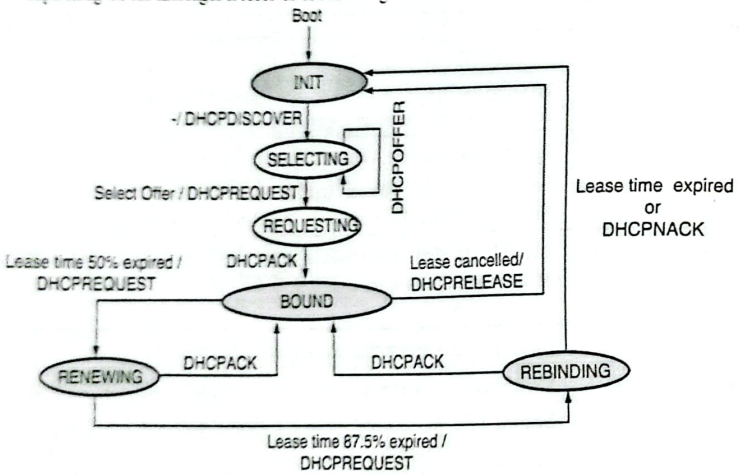


Fig. 1.21.4 : DHCP Transistion Diagram

- **INIT State:** When the DHCP client first starts, it is in the INIT state (initializing state). The client broadcasts a DHCPDISCOVER message (a request message with the DHCPDISCOVER option), using port 67.
- **SELECTING State :** After sending the DHCPDISCOVER message, the client goes to the selecting state. Those servers that can provide this type of service respond with a DHCPOFFER message. In these messages, the servers offer an IP address. They can also offer the lease duration. The default is 1 hour. The server that sends a DHCPOFFER locks the offered IP address so that it is not available to any other clients. The client chooses one of the offers and sends a DHCPREQUEST message to the selected server. It then goes to the requesting state. However, if the client receives no DHCPOFFER message, it tries four more times, each with a span of 2 seconds. If there is no reply to any of these DHCPDISCOVERs, the client sleeps for 5 minutes before trying again.
- **REQUESTING State :** The client remains in the requesting state until it receives a DHCPACK message from the server that creates the binding between the client physical address and its IP address. After receipt of the DHCPACK, the client goes to the bound state.
- **BOUND State :** In this state, the client can use the IP address until the lease expires. When 50 percent of the lease period is reached, the client sends another DHCPREQUEST to ask for renewal. It then goes to the renewing state. When in the bound state, the client can also cancel the lease and go to the initializing state.
- **RENEWING State :** The client remains in the renewing state until one of two events happens. It can receive a DHCPACK, which renews the lease agreement. In this case, the client resets its timer and goes back to the bound state. Or, if a DHCPACK is not received, and 87.5 percent of the lease time expires, the client goes to the rebinding state.
- **REBINDING State :** The client remains in the rebinding state until one of three events happens. If the client receives a DHCPNACK or the lease expires, it goes back to the initializing state and tries to get another IP address. If the client receives a DHCPACK, it goes to the bound state and resets the timer.

1.22 TELNET (TERMINAL NETWORK)

GQ. Explain in brief : Telnet.
GQ. Explain TELNET in detail with respect to Server and Client communication?

1. TELNET is an abbreviation for *TErminaL NETwork*. It is the standard TCP/IP protocol for **virtual terminal service** as proposed by the International Organization for Standards **(ISO).**
2. TELNET enables the establishment of a connection to a remote system in such a way that the local terminal appears to be a terminal at the remote system.
3. TELNET is a **general-purpose client/server application** program.
4. Telnet was developed in 1969.

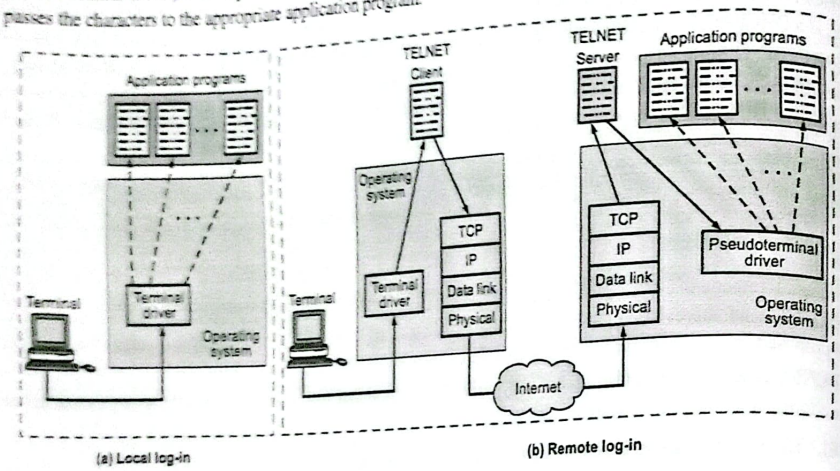
1.22.1 Timesharing Environment

- TELNET was designed at a time when most operating systems, such as **UNIX,** were operating in a timesharing environment. In such an environment, a large computer supports multiple users.
- The interaction between a user and the computer occurs through a terminal, which is usually a combination of keyboard, monitor, and mouse. Even a **microcomputer** can simulate a terminal with a **terminal emulator.**

1.22.2 Logging

1. In a **timesharing environment**, users are part of the system with some right to access resources. Each authorized user has identification and probably, a password. The user identification defines the user as part of the system.
2. To access the system, the user logs into the system with a **user id or login name.** The system also includes password checking to prevent an unauthorized user from accessing the resources. Fig. 1.22.1 shows the logging process.
3. When a user logs into a local timesharing system, it is called local **login.** As a user types at a terminal or at a workstation running a terminal emulator, the terminal driver accepts the keystrokes.
4. The **terminal driver** passes the characters to the operating system. The operating system, in turn, interprets the combination of characters and invokes the desired application program or utility. When a user wants to access an application program or utility located on a remote machine, she performs remote login.
5. Here the TELNET client and server programs come into use. The user sends the keystrokes to the terminal driver, where the local operating system accepts the characters but does not interpret them.

6. The characters are sent to the TELNET client, which transforms the characters to a universal character set called network virtual terminal (NVT) characters and delivers them to the local *TCP/IP* protocol stack.
7. The commands or text, in NVT form, travel through the Internet and arrive at the TCP/IP stack at the remote machine.
8. Here the characters are delivered to the operating system and passed to the TELNET server, which changes the characters to the corresponding characters understandable by the remote computer.
However, the characters cannot be passed directly to the operating system because the remote operating system is not designed to receive characters from a TELNET server.
9. It is designed to receive characters from a terminal driver. The solution is to add a piece of software called a ***pseudoterminal driver***, which pretends that the characters are coming from a terminal. The operating system then passes the characters to the appropriate application program.

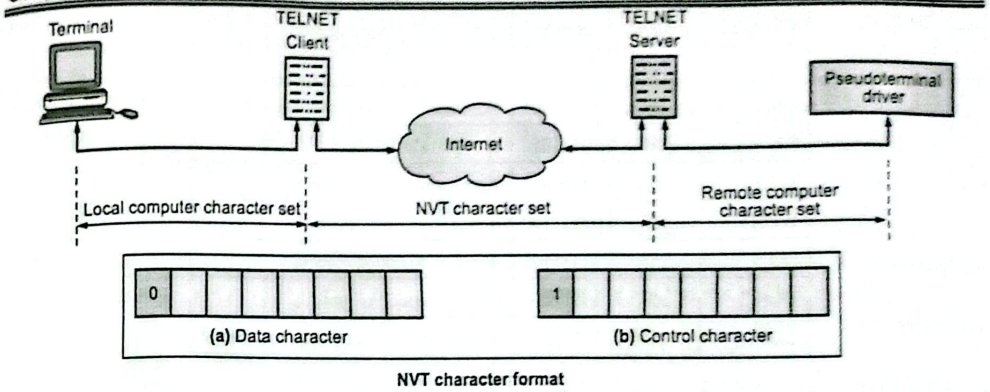


(1F28)**Fig. 1.22.1 : Local and remote log-in**

1.22.3 Network Virtual Terminal

GQ. Write short note on Network Virtual Terminal (NVT).

1. The mechanism to **access a remote computer** is complex. This is so because every computer and its operating system accept a special combination of characters as tokens.
2. For example, the end-of-file token in a computer running the DOS operating system is **Ctrl+z,** while the UNIX operating system recognizes **Ctrl+d**.
3. We are dealing with **heterogeneous systems**.
4. If we want to access any remote computer in the world, we must first know what type of computer we will be connected to, and we must install the specific terminal emulator used by that computer.
5. **TELNET** solves this problem by defining a universal interface called the **network virtual terminal (NVT)** character set. Via this interface, the client TELNET translates characters (data or commands) that come from the local terminal into NVT form and delivers them to the network.
6. The server TELNET, on the other hand, translates data and commands from NVT form into the form acceptable by the remote computer. Concept is shown in Fig. 1.22.2.



(1F29)**Fig. 1.22.2 : Concept of NVT**

7. NVT Character Set NVT uses two sets of characters, one for data and the other for control. Both are 8-bit bytes. For data, NVT is an 8-bit character set in which the 7 lowest-order bits are the same as ASCII and the highest-order bit is **O**.
8. To send control characters between computers (from client to server or vice versa), NVT uses an 8-bit character set in which the highest-order bit is set to **1**.

1.23 TELNET CONTROL FUNCTIONS

GQ. Explain Telnet Control Functions.

1. The telnet protocol includes a number of control functions. These are initiated in response to conditions detected by the client (**usually certain special keys or key combinations**) or server.
2. The detected condition causes a special character to be incorporated in the data stream.

Table 1.23.1 : Some NVT Control Character

Character	Decimal	Binary	Meaning
EOF	236	11101100	End of file
FOR	239	11101111	End of record
SE	240	11110000	Suboption end
NOP	241	11110001	No operation
DM	242	11110010	Data mark
BRIE	243	11110011	Break
IP	244	11110100	Interrupt process
AO	245	11110101	Abort output
AYT	246	11110110	Are you there?
EC	247	11110111	Erase character

Character	Decimal	Binary	Meaning
EL	248	11111000	Erase line
GA	249	11111001	Go ahead
SB	250	11111010	Suboption begin
WILL	251	11111011	Agreement to enable option
WONT	252	11111100	Refusal to enable option
DO	253	11111101	Approval to option request
DON'T	254	11111110	Denial of option request
IAC	255	11111111	Interpret (the next character) as control

1.23.1 Some Important Telnet Control Functions are as follows

1. **Interrupt Process :** This is used by the client to cause the suspension or termination of the server process. Typically, the user types **Ctrl-C** on the keyboard. An **IP (244)** character is included in the data stream.
2. **Abort Output :** This is used to suppress the transmission of remote process output. An **AO (238)** character is included in the data stream.
3. **Are You There :** This is used to trigger a visible response from the other end to confirm the operation of the link and the remote process. An **AYT (246)** character is incorporated in the data stream.
4. **Erase character :** Sent to the display to tell it to delete the immediately preceding character from the display. An **EC (247)** character is incorporated in the data stream.
5. **Erase line :** causes the deletion of the current line of input. An **EL (248)** character is incorporated in the data stream.
6. **Data Mark :** Some control functions such as **AO and IP** require immediate action and this may cause difficulties if data is held in buffers awaiting input requests from a (possibly misbehaving) remote process.

   To overcome this problem a **DM (242)** character is sent in a TCP Urgent segment, this tells the receiver to examine the data stream for "interesting" characters such as IP, AO and AYT. This is known as the telnet synch mechanism.

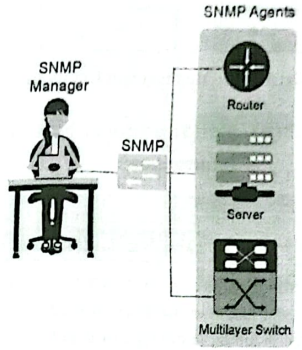
1.24 SNMP (SIMPLE NETWORK MANAGEMENT PROTOCOL)

GQ. Write short notes on the following : SNMP and MIB
UQ. Explain SNMP Protocol.
(SPPU- Q. 3(b), Oct. 16, 04 Marks)

A. Simple Network Management Protocol (SNMP)

1. **Simple Network Management Protocol (SNMP)** is an application-layer protocol used to manage and monitor network devices and their functions.
2. SNMP provides a common language for network devices to relay management information within single and multivendor environments in a local area network **(LAN) or wide area network (WAN).** The most recent iteration of SNMP, version 3, includes security enhancements that authenticate and encrypt SNMP messages as well as protect packets during transit.
3. One of the most widely used protocols; SNMP is supported on an extensive range of hardware from conventional network equipment like routers, switches and wireless access points to endpoints like **printers, scanners and internet of things (IoT) devices.**
4. In addition to hardware, SNMP can be used to monitor services such as **Dynamic Host Configuration Protocol (DHCP).** Software agents on these devices and services communicate with a network management system (NMS), also referred to as an SNMP manager, via SNMP to relay status information and configuration changes.
5. While SNMP can be used in a network of any size, its greatest value is evident in large networks. Manually and individually, logging into hundreds or thousands of nodes would be extremely time-consuming and resource-intensive.

6. In comparison, using SNMP with an NMS enables a network administrator to manage and monitor all of those nodes from a single interface, which can typically support batch commands and automatic alerts.
7. SNMP is described in the **Internet Engineering Task Force (IETF) Request for Comment (RFC) 1157** and in a number of other related RFCs.

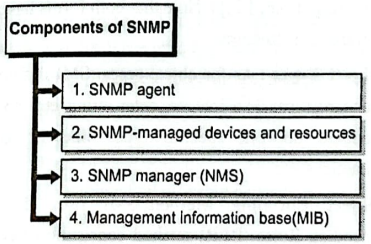


(1F12)**Fig. 1.24.1 : SNMP Configuration**

1.24.1 Components of SNMP

UQ. What are the components of SNMP ?
(SPPU- Q. 3(a), Dec. 16, 06 Marks, Q. 3(b), May 17, 04 Marks)

There are four main components in an SNMP-managed network :



(1F13)**Fig. 1.24.2 : Components of SNMP**

1. SNMP agent

(i) This program runs on the hardware or service being monitored, collecting data about various metrics like bandwidth use or disk space.

(ii) When queried by the SNMP manager, the agent sends this information back to the management system.

(iii) An agent may also proactively notify the **NMS** if an error occurs.

(iv) Most devices come with an SNMP agent preinstalled; it typically just needs to be turned on and configured.

2. SNMP managed devices and resources

These are the nodes on which an agent runs.

3. SNMP manager (NMS)

(i) This software platform functions as a **centralized console** to which agents feed information.

(ii) It will actively request agents to send updates via SNMP at regular intervals.

(iii) What a network manager can do with that information depends heavily on how feature-rich the NMS is.

(iv) There are several free SNMP managers available, but they are typically limited in their capabilities or the number of nodes they can support.

(v) At the other end of the spectrum, enterprise-grade platforms offer advanced features for more complex networks, with some products supporting up to tens of thousands of nodes.

4. Management information base (MIB)

(i) This database is a **text file (.mib)** that itemizes and describes all objects used by a particular device that can be queried or controlled using SNMP.

(ii) This database must be loaded into the NMS so that it can identify and monitor the status of these properties. Each MIB item is assigned an **object identifier (OID).**

1.24.2 Working of SNMP

GQ. Explain working of SNMP.

1. SNMP performs a multitude of functions, relying on a blend of **push and pull communications** between network devices and the management system.
2. It can issue read or write commands, such as resetting a password or changing a configuration setting.
3. It can report back how much bandwidth; CPU and memory are in use, with some **SNMP** managers automatically sending the administrator an email or text message alerts if a predefined threshold is exceeded.
4. In most cases, **SNMP** functions in a synchronous model, with communication initiated by the **SNMP** manager and the agent sending a response.

5. These commands and messages, typically transported over **User Datagram Protocol (UDP)** or Transmission Control Protocol/Internet Protocol (**TCP/IP**), are known as **protocol data units (PDUs)**.
   - **GET :** Generated by the SNMP manager and sent to an agent to obtain the value of a variable, identified by its **OID**, in an **MIB**.
   - **RESPONSE :** Sent by the agent to the SNMP manager, issued in reply to a **GET** request. Contains the values of the requested variables.
   - **GETNEXT :** Sent by the SNMP manager to agent to retrieve the values of the next **OID** in the MIB's hierarchy.
   - **GETBULK :** Sent by the SNMP manager to the agent to obtain large tables of data by performing multiple **GETNEXT** commands.
   - **SET :** Sent by the **SNMP manager** to the agent to issue configurations or commands.
   - **TRAP :** An asynchronous alert sent by the agent to the SNMP manager to indicate a significant event, such as an error or failure, has occurred.

1.24.3 Management with SNMP has Three basic Ideas

- A manager checks the agent by requesting the information that reflects the behaviour of the agent.
- A manager also forces the agent to perform a certain function by resetting values in the agent database.
- An agent also contributes to the management process by warning the manager regarding an unusual condition.

1.24.3(A) Management Components

UQ. What is the purpose of SMI and MIB in relation to SNMP. **(SPPU- Q. 1(a), Dec. 14, 06 Marks)**

UQ. What is the role of SMI and MIB in SNMP. **(SPPU- Q. 6(a), Oct. 16, 05 Marks)**

- Management is not achieved only through the SNMP protocol but also the use of other protocols that can cooperate with the SNMP protocol.
- Management is achieved through the use of the other two protocols:
  - SMI (Structure of management information) and
  - MIB(management information base).
- Management is a combination of SMI, MIB, and SNMP. All these three protocols such as abstract syntax notation 1 (ASN.1) and basic encoding rules (BER).

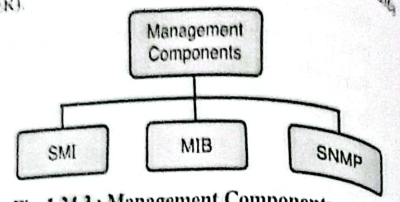


Fig. 1.24.3 : Management Components

1.24.3(B) SMI(Structure of Management Information)

UQ. What is the purpose of SMI and MIB in relation to SNMP ? **(SPPU - Q. 1(b), Dec. 19, 6 Marks)**

- The SMI (Structure of management information) is a component used in network management. Its main function is to define the type of data that can be stored in an object and to show how to encode the data for the transmission over a network.
- MI protocol is basically used to standardize the different object attributes like object Identifiers, object type and encoding methods for objects.
- **Object Identifiers:** Each SNMP object has a unique object identifier. SMI permits object identifiers to be either in name form (e.g. iso.org.dod.internet.mgmt.mib-2) or in number form (e.g. 1.3.6, 1,2,1). Note that both forms have a hierarchical structure.
- **Object Types :** As for object types, SMI defines both standard types like integer, octet string etc. and non-standard network specific types like IP address, Timeticks etc.
- **Object Encoding Method:** SMI uses the Basic Encoding Rules (BER) method consisting of type/tag, length and value, to encode objects along with their values, for transmission inside SNMP packets.
- SMI does not specify the list of objects for a specific network protocol/entity nor does it specify the type of objects to be used for a specific network protocol. These aspects are taken care by MIB.

1.24.3(C) MIB (Management Information base)

UQ. Explain MIB. **(SPPU- Q. 6(b), Dec. 16, 04 Marks)**

- The MIB (Management information base) is a second component for the network management.
- MIBs enable representation of device parameters in a standard format.
- MIBs are nothing but the actual set of objects supported by a network device, for controlling and monitoring by the SNMP protocol.
- These objects are classified and separately maintained in different MIB files. There would be a separate MIB file maintained by the SNMP agent on the network device, for each protocol/entity that can be managed by SNMP (e.g. System MIB, Chassis MIB, IP MIB, TCP MIB, UDP MIB, ICMP MIB, Interface MIB etc.).
- Each agent has its own MIB, which is a collection of all the objects that the manager can manage. MIB is categorized into eight groups: system, interface, address translation, ip, icmp, tcp, udp, and egp. These groups are under the mib object.

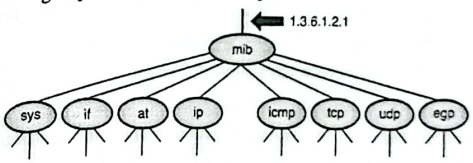


Fig. 1.24.4 : .MIB

- MIBs are organized in a tree like structure and each MIB variable has a unique object ID
- Each MIB file define three things, namely
  - List of objects supported for a specific protocol/entity
  - Type of each object
  - Hierarchical relationship between all the objects of a given protocol/entity
- For Example :
  - UDP MIB would contain objects like udpInDatagrams (denoting number of incoming UDP datagrams to the device), udpOutDatagrams (denoting number of outgoing UDP datagrams out of the device) etc.
  - The UDP MIB would also specify the type of these objects (for e.g. udpInDatagrams and udpOutDatagrams could be of type Integer). The UDP MIB could also contain a UDP Table consisting of the set of all current UDP socket connections handled by the device at that instant of time.
  - The entries in this table have a child relationship to the parent UDP table, which means that MIB files maintain a hierarchical relationship between the different objects pertaining to a specific network protocol/entity.

1.24.3(D) MIB Object Identifiers

UQ. Explain MIB with the help of its structure. **(SPPU- Q. 6(b), Aug. 15, 04 Marks)**

- Each object in the MIB has an object identifier (OID), which the management station uses to request the object's value from the agent.
- An OID is a sequence of integers that uniquely identifies a managed object by defining a path to that object through a tree-like structure called the OID tree or registration tree.
- When an SNMP agent needs to access a specific managed object, it traverses the OID tree to find the object.
- The MIB object identifier hierarchy and format is shown in the following Fig. 1.24.5.

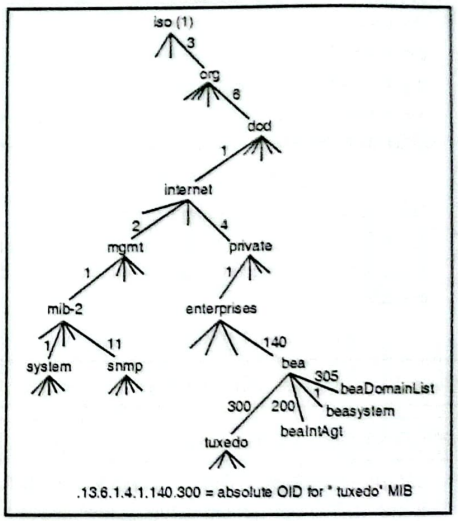


Fig. 1.24.5 : SNMP MIB Object Identifier Hierarchy and Format

- In this hierarchy, each BEA private MIB object that the BEA SNMP Agent software manages has a unique

object identifier. A prefix of .1.3.6.1.4.1.140 points to the objects in the BEA private MIB for the BEA SNMP Agent software.

(i) Absolute and Relative Object Identifiers

- Absolute OIDs specify a path to an attribute from the root of the OID tree. Absolute OID names always begin with a dot and must specify every node of the OID tree from the top-most node to the specific managed object. For example:

  .1.3.6.1.2.1.1.1

- Relative OIDs specify a path to the attribute relative to some node in the OID tree.
- **For example**, 2.1.1.1 specifies the sysDescr object in the system group, relative to the Internet node in the OID tree.

(ii) Specifying Object Identifiers

- In addition to using the "dot-dot" notation, a series of integers separated by dots to describe OIDs, you can also express OIDs by using textual symbols instead of numbers to represent nodes in the path to the object, or by using a combination of both integers and textual symbols.
- A symbolic OID uses mnemonic keywords to specify the managed object.
- **For example**

  mgmt.mib-2.system.sysContact

- The following numeric OID uses integers to specify the same managed object:

  2.1.1.7

- Note that 2.1.1.7 in this example is a relative OID.
- An OID can combine both symbolic and numeric representations of individual nodes of the OID tree; for example:

  mgmt.mib-2.1.sysContact

1.25 SECURE SHELL(SSH)

GQ. Explain What is SSH ?

- SSH stands for **Secure Shell** or **Secure Socket Shell**. It is one of the major protocols that is used in order to **access the network devices** and **servers over the Internet**. It was originally designed to replace **TELNET.**
- A **cryptographic network protocol** allows two computers to communicate and share the data over an insecure network such as the internet.
- It is used to login to a remote server to execute commands and data transfer from one machine to another machine.
- It is basically a network protocol, and it mainly runs on top of **TCP/IP protocol.**
- The SSH protocol was developed by **SSH communication security Ltd** to safely communicate with the remote machine.
- Secure communication provides a **strong password authentication** and **encrypted communication** with a public key over an insecure channel.
- It is used to **replace unprotected remote login protocols** such as Telnet, rlogin, rsh, etc., and insecure file transfer protocol FTP.
- This protocol mainly encrypts the traffic in both directions; with the help of this feature, you can prevent trafficking, sniffing, and password theft.
- By default, SSH runs on **Port number 22** and it can be changed. It is suitable for **Public Networks**. Its security features are widely used by network administrators for managing systems and applications remotely.

☞ **There are two versions of SSH**

1. SSH-1 2. SSH-2

These are totally incompatible. The first version, SSH-1, is now deprecated because of security flaws in it.

1.25.1 Components of SSH

GQ. Explain Components of SSH.

SSH is an application-layer protocol with three components, as shown in Fig. 1.25.1.

1. SSH Transport-Layer Protocol (SSH-TRANS)
2. SSH Authentication Protocol (SSH-AUTH)
3. SSH Connection Protocol (SSH-CONN)

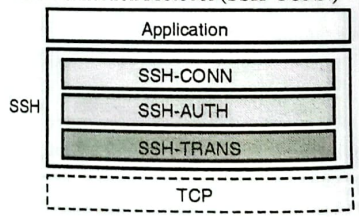


Fig. 1.25.1 : Components of SSH

1.25.1(A) SSH Transport-Layer Protocol (SSH-TRANS)

- Since TCP is not a secured transport-layer protocol, SSH first uses a protocol that creates a secured channel on top of the TCP. This new layer is an independent protocol referred to as SSH-TRANS.
- When the procedure implementing this protocol is called, the client and server first use the TCP protocol to establish an insecure connection.
- Then they exchange several security parameters to establish a secure channel on top of the TCP.
- The Transport Layer protocol part of the SSH mainly used to provide
  - The confidentiality of the data,
  - The server /host authentication, and
  - Data Integrity.
- Optionally it also provides data compression as well.
- **Server Authentication**
  - The Host keys are asymmetric in nature like public/private keys.
  - The server makes use of a Public key in order to prove its identity to the client.
  - Mainly the client verifies that contacted server is a "known" host with the help of the database that it maintains.
  - Once the authentication of the server is done then session keys are generated.
- **Session Key Establishment**
  - After the authentication of the server, the client and the server agree upon the cipher that is to be used.
  - The Session keys are usually generated by both the client as well as the server.
  - These keys are mainly generated before the user authentication so that usernames and passwords can be sent are encrypted.
  - The Session keys are generally replaced at regular intervals(we can say like an hour) during the session and then are destroyed immediately after use.
- **Data Integrity**
  - SSH mainly makes the use of Message Authentication Code (MAC) algorithms in the order of the data integrity check.

1.25.1(B) SSH Authentication Protocol (SSH-AUTH)

- After a secure channel is established between the client and the server and the server is authenticated for the client, SSH can call another procedure that can authenticate the client for the server.
- This layer defines a number of authentication tools similar to the ones used in SSL. Authentication starts with the client, which sends a request message to the server.
- The request includes the user name, server name, the method of authentication, and the required data.
- The server responds with either a success message, which confirms that the client is authenticated, or a failed message, which means that the process needs to be repeated with a new request message.
- For the authentication purpose there are several methods that can be used
  - Typed Passwords
  - Public-key authentication etc.

1.25.1(C) SSH Connection Protocol (SSH-CONN)

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

1.25.2 Working of SSH

GQ. Explain Working of SSH.

- The SSH protocol works in a **client-server** model, which means it connects a secure shell client application (End where the session is displayed) with the SSH server (End where session executes) shown in Fig. 1.25.2.
- It was initially developed to replace insecure login protocols such as Telnet, rlogin, and hence it performs the same function.

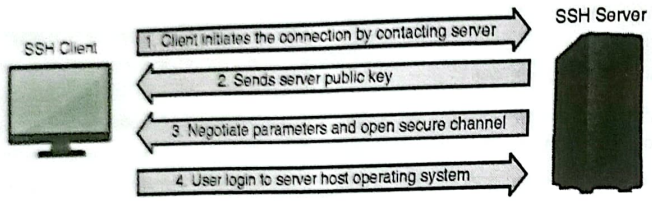


Fig. 1.25.2 : Working of SSH

- The basic use of SSH is to connect a remote system for a terminal session and to do this, following command is used:

ssh UserName@SSHserver.test.com

- The above command enables the client to connect to the server, named *server.test.com*, using the ID *UserName*.
- If we are connecting for the first time, it will prompt the remote host's public key fingerprint and ask to connect. The below message will be prompt:
  1. The authenticity of host 'sample.ssh.com' cannot be established.
  2. DSA key fingerprint is 01:23:45:67:89:ab:cd:ef:ff:fe:dc:ba:98:76:54:32:10.

  Are you sure you want to continue connecting (yes/no)?
- To continue the session, we need to click yes, else no. If we click yes, then the host key will be stored in the known_hosts file of the local system.
- The key is contained within the hidden file by default, which is **/.ssh/known_hosts** in the home directory.
- Once the host key is stored in this hidden file, there is no need for further approval as the host key will automatically authenticate the connection.

1.25.3 Applications

GQ. List and Explain Applications of SSH.

UQ. Explain Frame format of SSH packet. (SPPU- Q. 8(a), May 17, 08 Marks)

Although SSH is often thought of as a replacement for TELNET, SSH is, in fact, a general-purpose protocol that provides a secure connection between a client and server.

1.25.3(A) SSH for Remote Logging

Several free and commercial applications use SSH for remote logging. Among them, we can mention PuTTy, by Simon Tatham, which is a client SSH program that can be used for remote logging. Another application program is Tectia, which can be used on several platforms.

1.25.3(B) SSH for File Transfer

- One of the application programs that is built on top of SSH for file transfer is the Secure File Transfer Program (sftp). The sftp application program uses one of the channels pro- vided by the SSH to transfer files.
- Another common application is called Secure Copy (scp). This application uses the same format as the UNIX copy command, cp, to copy files.

1.25.3(C) Port Forwarding

- One of the interesting services provided by the SSH protocol is port forwarding. The secured channels available in SSH can be used to access an application program that does not provide security services.
- Applications such as TELNET and Simple Mail Transfer Protocol (SMTP), can use the services of the SSH port forwarding mechanism.
- The SSH port forwarding mechanism creates a tunnel through which the messages belonging to other protocols can travel. For this reason, this mechanism is sometimes referred to as SSH tunneling.
- Fig. 1.25.3 shows the concept of port forwarding for securing the FTP application.

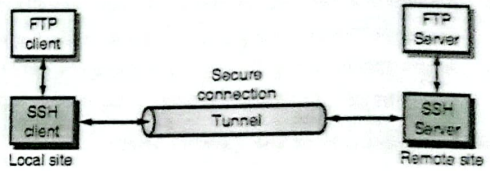


Fig. 1.25.3 : Port Forwarding

- The FTP client can use the SSH client on the local site to make a secure connection with the SSH server on the remote site.
- Any request from the FTP client to the FTP server is carried through the tunnel provided by the SSH client and server.
- Any response from the FTP server to the FTP client is also carried through the tunnel pro- vided by the SSH client and server.

1.25.4 Format of the SSH Packets

GQ. Explain the format of the SSH packets.

- Fig. 1.25.4 shows the format of packets used by the SSH protocols.
- The length field defines the length of the packet but does not include the padding.
- One to eight bytes of padding is added to the packet to make the attack on the security provision more difficult. The *cyclic redundancy check* (CRC) field is used for error detection.
- The type field designates the type of the packet used in different SSH proto- cols. The data field is the data transferred by the packet in different protocols.

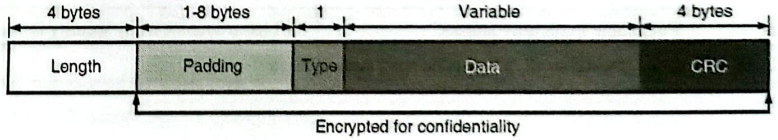


Fig. 1.25.4 : SSH Packet Format

1.25.5 Advantages and Disadvantages of SSH (Secure Shell)

GQ. List advantages and disadvantages of SSH.

Advantages of SSH (Secure Shell)

1. SSH protocol offers multiple services using the same protocol.
2. With the help of strong encryption, this protocol offers the privacy of the data of the user.
3. It is freely available.
4. It is used for non-commercial use.
5. It also allows the user to view the contents of directories, edit the files, and access the custom database applications remotely.
6. It is used to authenticate the identity of senders as well as receivers.
7. For simple VPNs tunneling of ports work in an effective way.
8. Allows the user to view the contents of directories, edit files, and access the custom database applications remotely.
9. The secure shell also helps to securely tunnel insecure applications like SMTP, IMAP, POP3, and CVS.

Disadvantages of SSH (Secure Shell)

1. This protocol does not help to protect from trojan horses or from viruses.
2. This protocol is only applicable to applications based on TCP and not applicable to applications based on UDP.
3. This protocol requires more technical knowledge.

1.25.6 Difference between SSH and Telnet

GQ. State difference between SSH and Telnet.

Table 1.25.1 : Comparison of SSH and Telnet

COMPARISON	TELNET	SSH
Security	Less secured	Highly secured
Uses port number	23	22
Data format	Telnet transfers the data in plain text.	Encrypted format is used to send data and uses a secure channel.
Authentication	No privileges are provided for user's authentication.	Uses public key encryption for authentication.
Suitability of network	Private networks are recommended.	Suitable for Public networks.
Vulnerabilities	Vulnerable to security attacks.	SSH has overcome many security issues of telnet.
Bandwidth Usage	Low	High

Chapter Ends...

❑❑❑

Unit II

CHAPTER 2 Wireless Standards

Syllabus

Wireless LANs: Fundamentals of WLAN, Design goals, Characteristics, Network Architecture, IEEE 802.11: components in IEEE 802.11 network, Physical Layer, MAC Sub Layers : DCF, PCF, Hidden and exposed station problem, Frame format, Addressing Mechanism, IEEE 802.15.1 Bluetooth: Architecture Layers, operational states, IEEE 802.16 WiMax: Services, Architecture, Layers, comparison between Bluetooth, IEEE 802.11 and IEEE 802.16.