## Ans 1 )

A to F :- Since A doesn’t know the MAC address of F , thus A broadcasts an ARP request (checking for i6) to find F’s MAC address. This request reaches S1 which floods it to all its ports except 1(A , from where the request came) . The request then reaches S2 , which again floods it (since it is still broadcast DST = ff:ff:ff:ff:ff:ff ) . Now the request reaches E,D,F out of which F responds with its IP (i6) and MAC(m6) and this reply is directly sent to A via S1 and S2 which stores it in the ARP table. Host A can now send the data packet directly to F using F's MAC address (m6). S1 sees that the destination MAC address is m6 and forwards it to S2 via Port 4.S2 sees that the destination MAC address is m6 and forwards the frame to Port 3 (connected to F). Host F receives the packet and stores A’s MAC address (m1) in its ARP table.

E to B :- This follows the same procedure as above with the ARP request being flooded from S2 to S1 and then to B.

S2 to C : - S2 broadcasts an ARP request which reaches S1 and then to A,B,C . Thus S2 learns that C is connected to port 3 and is reachable via port 4(through S1)

A to B :- Host A sends an ARP request to S1 which floods it and thus the request reaches B which respond to it with its MAC (m2) and A stores it in its ARP table and maps i2 to m2.

## Ans 2)

A to B :- A send ARP request with DST = ff:ff:ff:ff:ff:ff to R1 . Since B is connected to R1 , it forwards the request to B which replies to A with its MAC (mb) and A them maps ib to mb.R1’s ARP table now contains ma->ia and mb->ib . Since Host A and Host B are on different subnets, Router R1 acts as the gateway for Host A. The routing table of Router R1 is updated to reflect that to reach ib (Host B), packets must be forwarded to its **interface 2** (towards Host B).

B to C :- Same procedure as above with B sending its ARP request to R2 and then to C.

C to A :- C broadcasts an ARP request looking for A which reaches R2 and is then forwarded to R1 as R1 is the next hop towards A. Now even though R1 knows ia and ma it will not respond to the request as the ARP is specifically looking for the host with IP ia . Thus A sends ARP reply directly to C.

## Ans 3)

**HTTP** (Hypertext Transfer Protocol) :- HTTP is a stateless protocol, meaning each request is independent and does not retain information about previous requests.

Client (Browser) sends an HTTP request to the server for a specific resource (e.g., a webpage).

The serve**r** processes the request, fetches the resource, and sends an HTTP response back to the client.

The response usually includes an HTML document, along with any referenced media (like images, stylesheets, or scripts).

HTTP operates over TCP, typically using port 80.

**SMPT** (Simple Mail Transfer Protocol) :- SMTP is used for sending emails between servers. It is a push protocol used by email clients (like Outlook or Gmail) to send outgoing mail to mail servers, which then relay the mail to the recipient's mail server.

SMTP operates on a **client-server model** where the email client communicates with the outgoing mail server to send an email.

The client submits an email message, including the sender's and recipient's email addresses, subject, body, and attachments.

The email server processes the message and sends it to the recipient's mail server using SMTP.

SMTP is a push protocol because it is used to send mail, not to retrieve it. For retrieving mail, protocols like POP3 or IMAP are used.

It operates on port 25 (though modern implementations may use other ports such as 587 for submission).

**POP3** (Post Office Protocol version 3) :- POP3 is used by email clients to **retrieve emails** from a mail server. It allows users to download messages to their local devices for offline access. Once messages are downloaded, they are typically deleted from the server.

* When an email client connects to the mail server, it uses POP3 to download the emails stored on the server.
* POP3 has a simple **request-response** model. The client sends a request to the server to download new messages, and the server sends back the messages.
* Once the messages are downloaded, they are removed from the server by default, though some configurations allow them to be left on the server.
* POP3 works over **port 110**.

**Limitations**: POP3 does not support **synchronization** across multiple devices. Once emails are downloaded, they are only available on the device they were downloaded to.

**IMAP** (Internet Message Access Protocol):- IMAP is also used by email clients to retrieve emails from a mail server, but unlike POP3, IMAP allows for **synchronization** of emails across multiple devices, as messages remain stored on the server.

IMAP allows the client to **view** and manipulate messages directly on the server (e.g., mark as read, delete, or move to folders).

Emails are stored on the server, and the client interacts with them in real-time.

IMAP enables multiple devices to access the same email account, with all actions reflecting on the server and syncing across devices.

IMAP works over **port 143** (or **port 993** for secure communication via SSL/TLS).

**FTP** (File Transfer Protocol):- FTP is used for transferring files between a client and a server over a network. It allows users to upload and download files to/from a remote server.

FTP is a **client-server protocol**: the client initiates a connection to the server, usually by providing a username and password.

FTP operates in **two channels**:

* The **control channel** (for commands) is usually on **port 21**.
* The **data channel** (for file transfer) is dynamically allocated and may vary.

FTP supports **both active and passive modes**. In active mode, the client listens on a random port and the server connects to it. In passive mode, the server listens on a random port and the client connects to it.

**SSL** (Secure Socket Layer):- SSL is a security protocol designed to provide a secure channel for communication over the internet. It encrypts data sent between clients and servers to protect it from eavesdropping and tampering.

* SSL uses a combination of **symmetric encryption** (for data transfer) and **asymmetric encryption** (for key exchange).
* When a client connects to a server, the server provides a **certificate** to the client, which contains a public key.
* The client uses the public key to encrypt a secret key (for symmetric encryption). Both parties now use this secret key to encrypt and decrypt the data sent between them.
* SSL operates over a **TCP connection** and is typically used in **HTTPS**, **FTPS**, **SMTP over SSL**, and other protocols.

**TLS** (Transport Layer Security):- TLS is the successor to SSL and provides secure communication over a computer network. It encrypts the data exchanged between the client and server to prevent eavesdropping, tampering, and forgery.

* TLS operates similarly to SSL but with stronger encryption algorithms and improved security features.
* A typical TLS handshake involves the client and server agreeing on a cipher suite, authenticating each other, and establishing a shared secret key for symmetric encryption.
* TLS is widely used in **HTTPS**, **email protocols (SMTP, IMAP, POP3)**, **VPNs**, and more.
* TLS uses **port 443** for secure web traffic (HTTPS).

**HTTPS** (Hypertext Transfer Protocol Secure):- HTTPS is the secure version of HTTP, where communication between the client (browser) and the server is encrypted using SSL or TLS.

* HTTPS ensures that all data transferred between the client and server is encrypted, protecting sensitive information like login credentials, credit card numbers, etc., from eavesdropping and tampering.
* When a user accesses a website via HTTPS, the browser first establishes a secure connection using the **TLS/SSL handshake**.
* The website server must present a **digital certificate** (SSL/TLS certificate) to authenticate its identity and ensure that it is the legitimate server.
* HTTPS operates over **port 443**.

## Ans 5)

**User Device**: Initiates requests, sends credentials, and receives responses.

**DHCP Server**: Assigns the user device an IP address when connecting to the network.

**DNS Server**: Resolves domain names (e.g., www.google.com) to IP addresses.

**Web Server**: Hosts the website and serves web content (pages, images, etc.).

**Authentication Server**: Verifies user credentials during login.

**Database Server**: Stores and retrieves user data (e.g., login credentials, profile data).

**ISP**: Routes traffic between the user device and other internet services, ensuring internet connectivity.

User Device (Client)

|

v

Types URL (e.g., www.google.com)

|

v

DNS Query to DNS Server

|

v

DNS Server Resolves IP Address of Web Server

|

v

User Device Connects to Web Server

|

v

Web Server Processes Request (e.g., /login)

|

v

Web Page Sent to User Device (HTML, CSS, JS)

|

v

User Submits Login Credentials (Username, Password)

|

v

Web Server Sends Credentials to Authentication Server

|

v

Authentication Server Verifies Credentials

|

v

Authentication Server Queries Database Server (if needed)

|

v

Database Server Returns User Data (Optional)

|

v

Authentication Server Returns Session Token to User Device

|

v

User Device is Logged In and Accesses Personalized Content

|

v

ISP Routes All Traffic Between User Device and Servers