Contents

Unit 3

The Application Layer

Structure of the Unit

3.1 Unit Outcomes

After the successful completion of this unit, the student will be able to:

- Examine conceptual and implementation aspects of network applications.
- Explain the working of the File Transfer protocol.

3.2 Introduction

There are many useful applications being created on the Internet today for users at home, businesses, educational institutions, government organizations, banks, and many more. In the 1970s and 80s, only text-based applications like email, file transfer, newsgroups, and accessing computers remotely were some of the popular applications. In the 1990s, the creation of the World Wide Web (WWW) allowed applications like web surfing, searching, and e-commerce to be developed. From 2000 onwards, multimedia applications like voice-over-IP (VoIP) and video over IP such as Skype, video distribution by users such as YouTube, movies on demand such as Netflix, online gaming applications, etc were made available.

In the layered architecture of computer networks, every layer uses the services provided by the layer below it and also provides services to the layer above it. But as the application layer is the topmost layer, it only uses the services provided by the transport layer (below it) provides services to the application programs running here, and directly communicates with the users. Between the two application layers of the sender and the receiver, there is a logical connection through which they send messages to each other.

Let us understand how communication takes place at the application layer through a scenario. Consider a situation in which a research scientist working at India Research, orders a book online from Navbharat Books. Communication between Jay and Ram happens at five different levels following the TCP/IP implementation model as discussed in Section 2.2.

In Fig.3.1, Jay's host in the Indian Research company creates a logical connection with Ram's host and gives the feeling of a physical connection between them. The actual communication between Jay and Ram happens through routers R2, R4, R5, and R7.

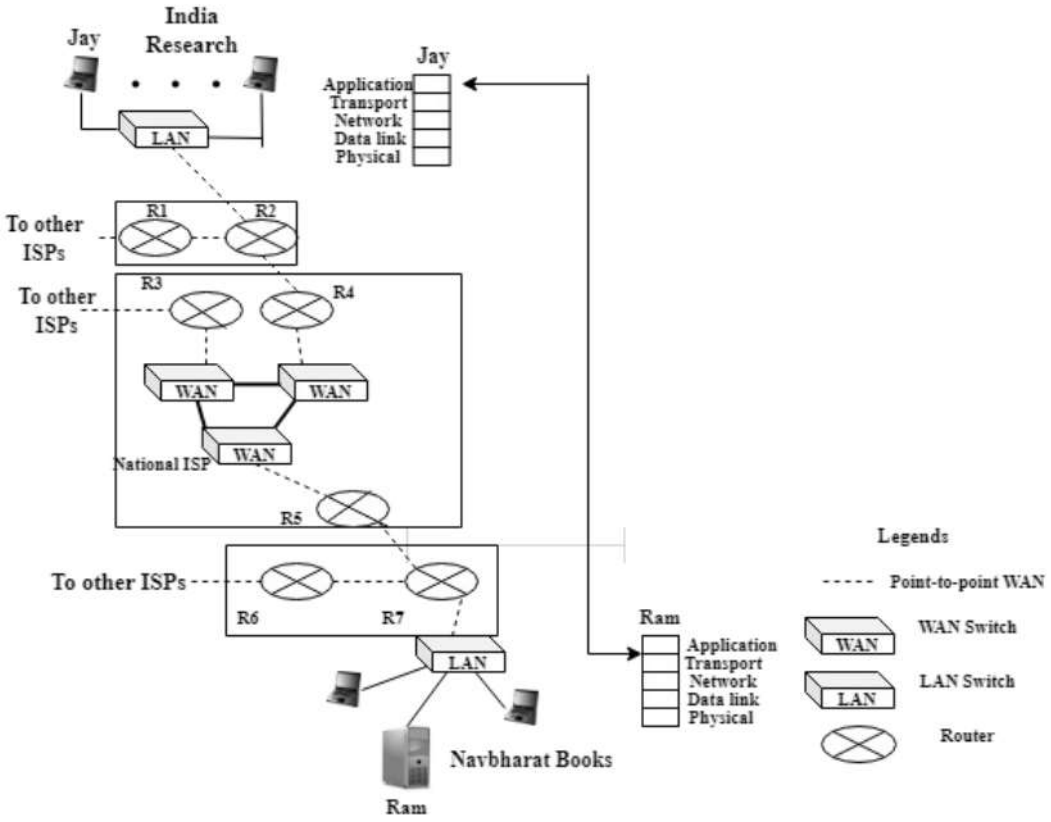

Fig. 3.1 Logical Connection at the Application Layer

NOTE: The communication at all the layers above the physical layers is only logical.

The topics discussed in the following sections are:

- Principles of network applications.
- Network services required by the applications.
- The popular protocols serving the applications are Hyper Text Transfer Protocol (HTTP), and the File Transfer Protocol (FTP).

3.3 Principles of Network Applications

First of all, the applications on the Internet only run on the end systems and not on the networking

devices such as switches and routers. Also, all the end systems are different. So the important issue to be considered for network application development is writing programs that run on these different systems and communicate with each other through the networking devices.

3.3.1 Network Applications Architectures

The five-layered network architecture (Section 2.2.2) is different from the application architecture. The application developer sees the network architecture as a fixed entity providing a set of services to the applications. For an application developer, there are two modern application architectures or paradigms: Client Server and Peer-to-peer (P2P).

3.3.1.1 Client-Server Network

It is a Network Computing Model, where many Client systems or nodes are connected to a Server system. The client can be a device or a program. The client requests services from the Server and the Server responds. The clients can be web browsers, laptops, smartphones, desktops, etc.

The Server is often a high-end computing system offering services such as files, databases, web pages, or any shared resource such as a printer, a scanner, etc. This model is beneficial in the way that the Server controls access to its services and their security. But is expensive to set up and maintain and the operations get disrupted if the server fails. So the server should always be ON and running.

Fig.3.2 shows a client-server network.

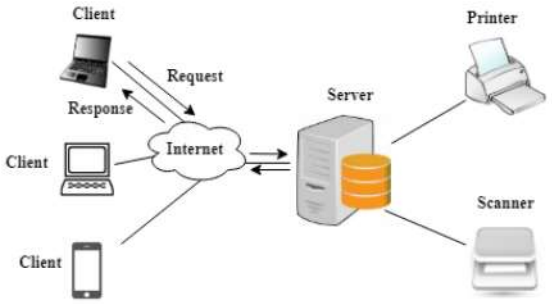

Fig.3.2: Client Server Network

3.3.1.2 Peer-to-Peer network

It is a simple network of computers, where each computer acts as a node for file sharing and other resources such as printers and scanners within the formed network. Here each node acts as a server and shares an equal workload.

There are three important benefits of a peer-to-peer network: network does not fail easily as one system failure does not bring down the network, highly scalable as adding new peers is easy and a quick process, very fast in operation, especially during file sharing a file can be stored on multiple systems and accessed.

Fig.3.3 shows a peer-to-peer network, with only 5 nodes connected in a Mesh topology. All the nodes can act as a server or a client.

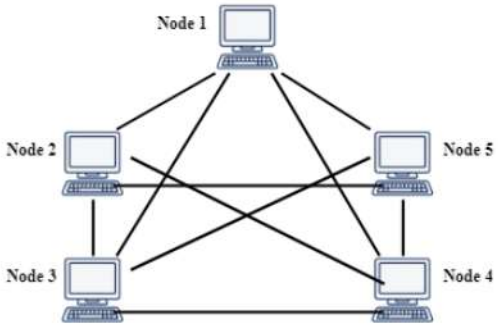

Fig. 3.3: Peer-to-Peer Network

3.3.1.1 Comparison of client-server and peer-to-peer architectures

Table 3.1 compares the two architectures:

Table 3.1: Comparison between client-server and peer-to-peer architectures

Parameter	Client-server	Peer-to-peer
Connectivity	Many clients are connected to one server.	Every node can act like a client a server
Cost of setup	Expensive to set up the server as it is built to run constantly to avoid network collapse.	Less expensive
Type of programs running on the nodes	Two different programs, one on the server and one on the client.	Similar or identical programs on the two hosts participating in the communication process.
Type of communication	Client programs only can request, and the server responds. Client hosts cannot communicate with each other.	Both hosts can request and respond. All the peers can communicate with each other.
ON status of the nodes	The server is always ON, to be able to service requests from clients.	The hosts are intermittently connected and are called peers.
Ownership	Servers owned by service providers. Groups of servers used to serve large numbers of clients are called data centers.	Peers not owned by service providers.
Addressing	The server has a fixed IP address.	No server
Scalability	Highly scalable without loss of performance	Not scalable without loosing performance
Security	More secure	Less secure
Purpose	Built for sharing of different resources like files, printers, scanners, etc	Built for connecting peers.
Applications	Web, Telnet, FTP, and email	File sharing (BitTorrent), Internet telephony (Skype)

3.3.2 Communicating Processes

The processes within a single-end system communicate with each other using Inter-process Communication (IPC) techniques which are features provided by the operating system software. However, in a computer communication network, the processes running on different end systems communicate at the application layer by exchanging messages (Fig. 3.1).

Within a communication session between two processes, usually, the process which starts the session, or contacts another process is called a client and the process which responds or waits to be contacted is called a server.

On the web, the browser process is the client and the web server is the server process. Whereas, in a P2P application where files are transferred, a process that uploads the file is called a server and the one which downloads the file is called the client.

3.3.2.1 The interface between a Process and the Computer Network

When process-to-process communication happens between two end systems, the messages sent and received pass through the network. The endpoints where the message from the sending process gets into the network and the receiving message passes from a network into a receiving process are called sockets. These are software interfaces as shown in Fig. 3.4

A socket acts as an interface between the application and transport layers also called an Application Programming Interface (API).

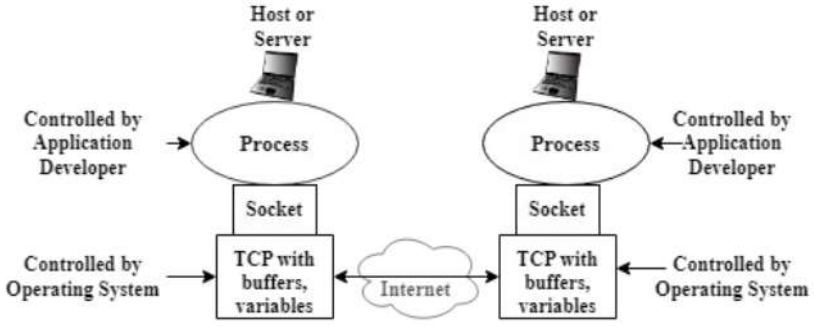

Fig. 3.4: Application Processes, sockets, underlying transport protocol

The application developer has control of all the different parameters on the application layer side of the socket, but can control only two parameters on the side of the transport layer as given below:

1. The transport protocol to be used
2. The maximum buffer and segment sizes

Using the other services provided by the transport layer, the required application is built.

3.3.2.2 Addressing Processes

When messages are sent from one host to another host, addressing a receiver process is required. This address consists of two pieces of information:

1. The address of the receiver on the internet.
2. A port number specifying the receiving process or destination port number.

3.3.3 Transport Services Provided by the Internet to Applications

For any application, there are two transport layer protocols to choose from and this is done based on the

specific needs of the applications. The services provided by the transport layer to the applications invoking it can be categorized based on: reliable data transfer, throughput, timing, and security.

3.3.3.1 Reliable data transfer

Applications needing reliable data transfer like e-mail, file transfer, web document transfer, and applications related to financial documents transfer can use TCP, and applications that are less tolerant to data loss such as multimedia applications can use UDP protocols at the transport layer.

3.3.3.2 Throughput

This parameter measures the number of bits/sec or the data rate at which the sending process can deliver to the receiving process. This rate depends on the available transmission bandwidth. As the transmission link is shared by the other sessions and the load keeps changing with time. This results in fluctuations in the throughput over time.

Applications that require a guaranteed throughput are called bandwidth-sensitive applications and those which can manage with less throughput are called Elastic applications.

Multimedia applications are bandwidth-sensitive applications, and email and web transfers are examples of elastic applications.

Many times multimedia applications use appropriate encoding techniques to adjust with the currently available throughput.

3.3.3.3 Timing

For real-time applications, the time delay in the transmissions should be less compared to non-real-time applications. The transport layer protocols can provide timing guarantees such that every bit of data pushed on the socket by the sender reaches the receiver side socket within a small time. For any application, a lower delay is always preferable.

3.3.3.4 Security

The transport layer can provide security features such as confidentiality, data integrity, and authentication of the sender at the endpoint for the applications. Confidentiality between the sending and the receiving process can be provided by encrypting the data at the transmitter and decrypting the data at the receiver.

Encryption is a process where a plaintext or message is converted to ciphertext, an unreadable, scrambled, or meaningless message so that the intruder cannot read the confidential data in transit. Similarly, decryption is a process where the ciphertext is converted to plaintext. Fig.3.5 shows this scenario.

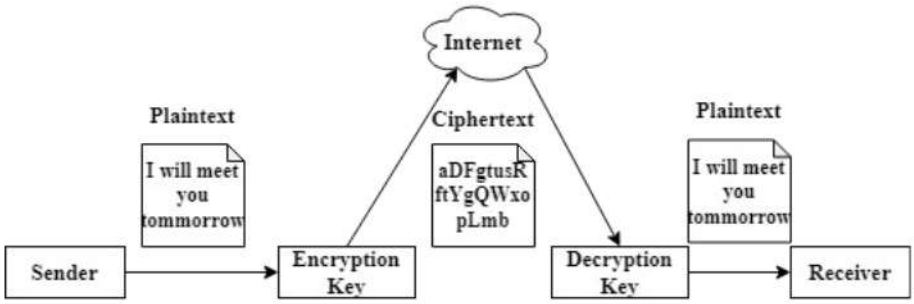

Fig. 3.5: Encryption and Decryption

There are many encryption algorithms available each of which is used based on the application and the degree of security. Data can be encrypted when it is stored at a place or when it is in transit on the internet.

Encryption uses a set of mathematical values called cryptographic keys. These key values or keys are agreed upon by the sender and the receiver. The keys should be complex enough for the intruder(third party) to guess so that breaking the cipher or decrypting is difficult.

There are two main types of encryption techniques:

- Asymmetric Encryption or public-key cryptography
  Here the same key is used for both encryption and decryption. It is simple and used where third-party attacks are fewer.
- Symmetric Encryption or private-key cryptography
  Here Encryption is performed with a help of a public key, whereas decryption is done with a secret key or private key.

For understanding encryption and decryption processes, let us consider one of the earliest and simplest encryption techniques known as Caesar Cypher. It was used by Julies Caesar for his military operations.

It uses the substitution method, where the plain text alphabets are replaced by other alphabets or symbols. In Caesar Cypher, each letter is replaced by a letter that is three places ahead of it in the alphabetic sequence.

Table. 3.2 shows the capital letters of the English alphabet and the numbers showing their position.

Table. 3.2: English Alphabets and their positions in the alphabetic sequence

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
13	14	15	16	17	18	19	20	21	22	23	24	25
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Algorithm:

- For each plaintext letter represented by the letter ‘p’ substitute it with the cyphertext letter represented by the letter ‘C’.
- Any encryption algorithm takes two inputs, first the plaintext and second the key.
  The ciphertext C = E (p, k) mod 26 = (p + k) mod 26,
  Where k is the key, and for Caesar Cypher k=3 (three positions ahead)

As the letters are in English the total number of alphabets is 26 and to maintain the range modulo 26 is used.

- The decryption algorithm also takes two inputs, first the Ciphertext and second the key. The plaintext p = D (C, k) mod 26 = (C - k) mod 26.

Example: Encrypt and decrypt the message " HELLO " using Caesar Cypher.

Solution: Referring to Table 9.2, Cipher for H = (7 + 3) mod 26 = 10
Cipher for E = (4 + 3) mod 26 = 7
Cipher for L = (11 + 3) mod 26 = 14
Cipher for L = (11 + 3) mod 26 = 14
Cipher for O = (14 + 3) mod 26 = 17

Substituting the plaintext letters with the letters obtained at the positions above gives the ciphertext as "KHOOR"

For decryption, Plaintext for K = (10 – 3) mod 26 = 7
Plaintext for H = (7 – 3) mod 26 = 4
Plaintext for K = (14 – 3) mod 26 = 11
Plaintext for K = (14 – 3) mod 26 = 11
Plaintext for K = (17 – 3) mod 26 = 14

Substituting the Cipher letter with the letters obtained at the positions above gives the plaintext as "HELLO"

Secure Socket Layer (SSL)

The basic TCP and UDP protocols do not provide encryption and decryption features, but an enhancement to the existing TCP called Secure Sockets Layer (SSL) has been developed by the internet community. SSL is not another transport layer protocol, but an enhancement implemented at the application layer. If an application wishes to use the SSL services, then the client and server sides of the application should include an SSL code with a set of libraries and classes. The SSL code can be used to encrypt and decrypt through its Application Program Interface (API). With SSL, data integrity and end-point authentication services can be provided.

3.4 Application Layer Protocols

The application layer protocols are responsible for structuring, timing, and adding fields to the messages which are sent by the processes for communicating with remote processes. These protocols define the following:

- Message types, syntax, and semantics
- All the rules regarding the time and the methods used by processes for sending and receiving the messages.

Some of the application-layer protocols are available in the public domain such as the Web's application-layer protocol HTTP, so if the HTTP client and servers follow the rules of the HTTP RFC,

then the browser program on such clients can retrieve the webpages from any of these web servers.
Other application-layer protocols which are private and proprietary are not available in the public domain, such as Skype.
Application-layer protocols are only a small part of the network applications, i.e a network application consists of many components.
Examples of network applications and their components:

- A web application (client-server application) used to fetch web pages as required.
  It consists of,
  - A document format such as a Hyper Text Markup Language (HTML).
  - Different browsers such as MS Explorer, Google Chrome, or Firefox.
  - Web servers such as Microsoft or Apache servers.
  - Application-layer protocols like the HTTP to define the format and the message sequences to be exchanged between the browser and the server.
- An e-mail application
  It consists of,
  - Mail servers containing user mailboxes
  - Mail client programs that allow the users to create and read messages.
  - Standard for the structure definition of the message.
  - Application-layer protocols like the Simple Mail Transfer Protocol (SMTP) to define the message transfers between the servers, between the client and the server, and interpret the headers of the message.

3.4.1 The Web and the HTTP

The Web or the World wide web (WWW) is an Internet application developed in 1994. It allows users or clients to retrieve information from the servers on demand. Users can publish their information on the web anytime at a very low cost. Navigating through the websites is possible through the use of hyperlinks and search engines. The information with a lot of graphics enhances the user experience. And users can interact with the web pages and websites using devices, forms, Java Applets, and Java scripts.
Some popular applications on the web are Facebook, Gmail, YouTube, etc.

Hyper Text Transfer Protocol (HTTP)

This protocol is implemented in the form of two components or programs: Client and Server running on different systems and communicating through HTTP messages.

A web page consists of entities called objects or files that are individually addressable by a single address called a Uniform resource locator (URL).
Examples of objects: An image, a video clip, an HTML file, or a Java Applet

A web page consists of a base HTML file with many referenced objects. The base file references the other objects with their URLs.

Eg: An HTML text file with four images has five objects in all.

A URL consists of two parts: The hostname of the server which contains the object and the path name of the object.
Example of URL: http://www.kanishkaSchool.edu/englishDepartment/picture1.gif
Here, www.kanishkaSchool.edu is the hostname, and /englishDepartment/picture1.gif is the path name of the object.

HTTP defines the way a client requests web pages from a server and also the way the server transfers the requested pages to the client. The request-response concept is illustrated in Fig. 3.6.

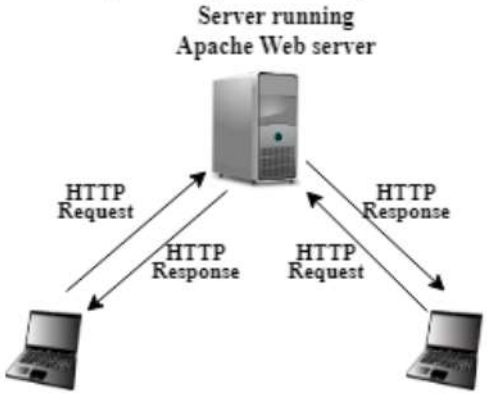

Fig. 3.6: HTTP request-response behavior

HTTP uses TCP as the transport layer protocol. The steps in the communication between an HTTP client and server are given below:

- The TCP connection between the client and server is established from the initiation of the client.
- Through their socket interfaces, the server and the browser processes access the TCP.
- HTTP request messages are sent by the clients into its socket interface.
- The server receives the request messages and sends response messages through its socket interface.
- The messages out of the socket interfaces are in the control of TCP, which transfers them reliably over the network with the help of other protocols running at the layers below it.

The HTTP server does not keep any state information of the clients during the requests, so it is called a stateless protocol.

3.4.1.1 Non-Persistent and Persistent Connections

These are the two options available for an application developer to use in regard to how the interaction between the client and server takes place. The interaction can happen with a series of back-to-back requests over separate request/response TCP connections or all of them happening on the same connection. The first one is called a non-persistent connection and the second one is called a persistent connection.

Non-persistent connection - Here, every TCP connection is closed after the server responds with an

object. So one TCP connection transports one request and one response message. So for a web page with 10 objects, there will be 11 TCP connections. The first response will be the base HTML file with 10 references to the objects. Also, users can configure their browsers to use either a serial connection or any number of parallel connections.

The time needed to request and receive an HTML file in a Non-persistent connection:

This can be demonstrated with the help of Fig.3.7

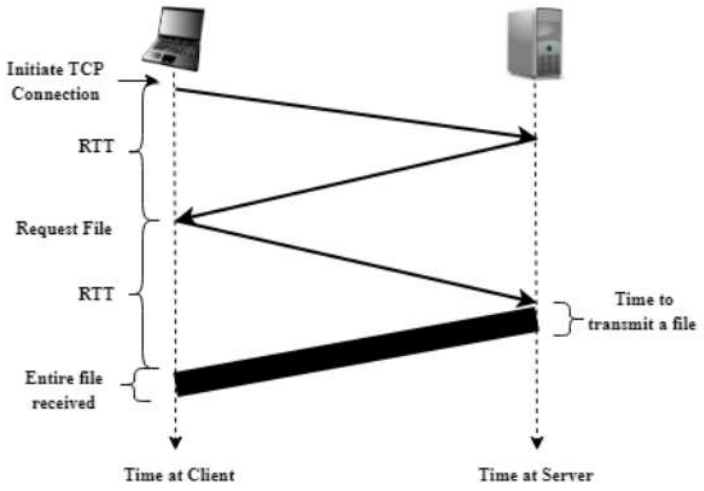

Fig. 3.7: Time needed to request and receive an HTML file

The total time needed for the request and response can be calculated as,

Two round trip times (RTTs) for sending two requests, one for connection and one for the file + transmission time at the server.

Limitations of Non-persistent connection:

- For every requested object a connection is established and maintained and this requires TCP buffers to be allocated and TCP variables to be maintained on both the client and the server sides.
- Each object delivery needs two RTTs for request and response.

Persistent connection – Here, the TCP connection is left open by the server after sending the first response, and the next request-response sessions can happen over the same connection. So a web page with 10 objects will be received over a single persistent TCP connection.

Requests for objects can be done serially one after the other by a client without waiting for the pending replies from the server. This feature is called pipelining and is programmable by the application developer. The connection is closed by the HTTP server if it is not used for a certain time interval which is also configurable.

3.4.1.2 HTTP message format

There are two types of HTTP messages: Request and Response and both messages consist of three parts.

The request message consists of,

- An initial request line consisting of three fields:
  - Method
  - URL of the resource
  - HTTP version
- Zero or more headers
  - Provide additional information about the type of data which the client will accept
- An optional message body
  - Holds the data to be sent to the server

Fig. 3.8 shows the general format of an HTTP request message.

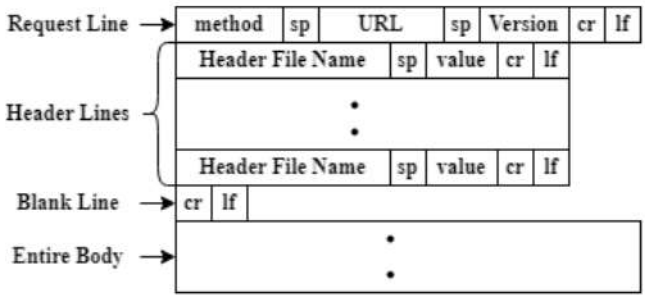

Fig. 3.8: General Format of an HTTP request message

Example of a request message:

GET /onedir/page1.html HTTP/1.1 --- request line
Host: www.myschool.edu --- header giving the hostname where the object resides
Connection: close --- header informing server to close the connection after sending the object.
Accept-language: fr ---- indicates that the user wants the object to be in the French language

Here the request line can be parsed as follows:

- GET is the method used when the browser requests the object.
- onedir/page1.html is the URL of the object requested by the browser
- HTTP/1.1 is version 1.1 of the HTTP

The response message consists of,

- An initial status line consisting of three fields:
  - Protocol version
  - Status code
  - Status message
- Zero or more headers
  - Provide additional information about the type of data being sent
- An optional message body
  - Holds the data to be sent to the client

Example of a response message:

HTTP/1.1 200 Ok --- request line

Connection: close --- header informing the client that connection would be closed after sending the object.

Date: Mon, 09 April 2023 15:22:05 GMT – indicates the date and time when the server created and sent the message

Server: Apache/2.2.3 --- server used to generate the message

Last Modified: Mon, 09 April 2023 15:10:04 GMT --- indicates the date and time when--the object and last modified.

Content-Length: 6821--- the size of the object given in a number of bytes.

Content -Type: text/html --- the type of the object.

(data …….) --- the data or content of the object or an HTML document sent in the response message.

3.4.1.3 Commonly used HTTP methods

Some commonly used HTTP methods are GET, PUT, POST, DELETE, and CONNECT. These methods are specified in uppercase letters.

GET: A request method is used to extract information from a server whose URL is provided in the header.

PUT: A request method used to create a new resource or replace the content with the resource it carries within this message.

POST: A request method used by a client to send data to a server using HTML forms.

DELETE: A request method used to delete a specified resource.

CONNECT A request method used to create a connection tunnel to a server whose URL is specified.

3.4.2 Cookies for User-Server Interaction

An HTTP Cookie is also known as Internet Cookie, Web Cookie, or Browser Cookie. A cookie is a piece of data created by a web server and sent to the user's browser for every session. These cookies are stored in the browser and when the user sends a request to the same server, the cookies will be sent along with the request. Thus, cookies help the server identify the requests coming from the same browser and help in remembering the stateful information as HTTP is a stateless protocol.

For Eg: When a user logs in to his email account, the browser sends the cookie information to his email server which helps the server to identify the user throughout the session when the user is logged in. So a cookie helps to create a user session layer on top of the stateless HTTP protocol.

Cookies are used by a server for three purposes:

1. Session management
    - Identifying user logins, saving shopping cart data or game scores, etc
2. Personalization
    - To remember user preferences, themes, and other user settings.
3. Tracking

- Used to record user behavior and analyze them.

3.4.2.1 Components of a Cookie

Cookie technology has four components (Refer to Fig.3.9)

1. A cookie header line in the HTTP response message.
2. A cookie header line in the HTTP request message.
3. A cookie file maintained at the client /user's side in the browser.
4. A back-end database on the Web Server.

3.4.2.2 Creation and Working of Cookies

This can be understood with the help of an example as shown in Fig. 3.9.

It is assumed that a user at the client host accesses the web through his browser and sends an HTTP request to the Amazon website for the first time and has already visited the eBay website earlier.

On this request, the server creates a unique identification number (ID:1678) and an entry into its backend database. It then sends a "Set-cookie" header along with the created ID.

When the user's browser receives the HTTP response message with the Set cookie header, it saves it in a special cookie file that it manages. The information stored includes the hostname of the server and the ID in the Set-cookie header.

The cookie file also has an entry for the eBay website. When the user browses the Amazon website later, his browser consults the cookie file and extracts the ID for this site, and appends a cookie header line along with the usual HTTP request. So, the Amazon server tracks the user's activity but it does not know the user's name. Based on the cookie ID, the server can find out which pages the user visited, the order of the pages visited, and the time of visit. This can help the server to recommend the products to the user and any products that are in the cart already will be shown. Also, if the user has already registered with the website, the user need not provide his personal details again as the server associates the cookie ID with the details.

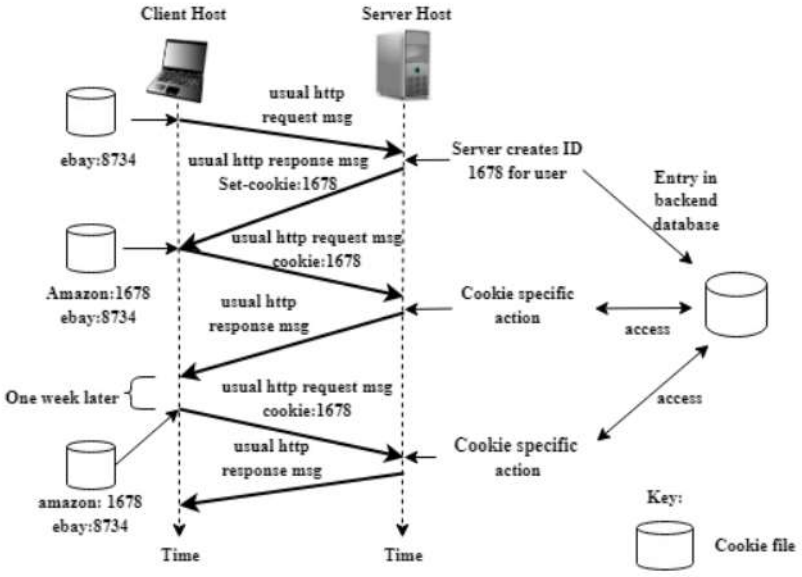

Fig. 3.9: Keeping user states with cookies

3.4.3 Web Caching

A Web cache is also known as a Proxy server. So, it is a server working on behalf of another origin server. This server has its own disk storage to save all the recently requested web pages or objects. So any requests from clients to the Origin server will be directed to the Proxy server first and if the requested page is unavailable, the request is sent by the proxy server to the origin server (Refer to Fig.3.10). For this operation to be performed, the user's browser has to be configured.

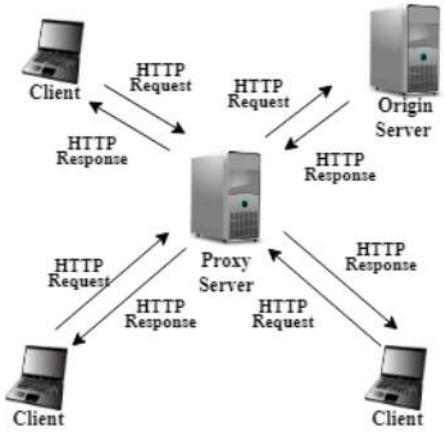

Fig. 3.10: Clients requesting objects through a Web cache

Let us understand this operation with an example, where a browser requests an object http://www. Schoolx.edu/campus.gif

1. The browser of a user establishes a TCP connection with the Proxy server and then sends an HTTP request to it.
2. The Web server checks for the requested object and if available responds back with the requested object else the proxy server requests the Origin server for the requested object on behalf of the client/user.
3. The Origin Server responds with the web page to the proxy server, which then saves a copy of the page into its disk storage and sends a copy to the requested client browser within an HTTP response message.

So in the above discussion, we note that a Web cache acts both like a web server and a client whenever a requested web page is not available in it.

A web cache is bought and installed by an Internet Service Provider.

Eg: There can be a university web cache installed in its network with all the campus browsers configured to be directed at it.

3.4.3.1 Use of Web Cache

There are two important benefits of using a Web cache:

1. Most of the time there is a high-speed connection between the client and the proxy server and if the object requested by the client is available in the proxy server, then the response time for the client request will be very small.
2. Web caches can reduce the traffic on an institution's access link to the Internet. This reduces the cost of the system, as the institution does not have to upgrade its bandwidth more often.

3.4.4 The Conditional GET

The deployment of the web cache or the proxy server reduces the response time for the users but introduces a different issue which is when a requested object copy in the cache is stale or the object has been modified in the Origin web servers after it has been copied into the cache. So in this situation, the new copy of the requested object has to be sent to the client or the user.

So, to solve the above issue, HTTP has a mechanism to verify whether the requested object has been modified in the Web server. This mechanism is called the conditional GET.

It is called conditional GET if,

1. The request message uses the GET method.
2. The request method includes an IF-Modified-Since header line.

Let us understand this mechanism through an example.

1. A client sends a requests message for a specific to server:
   GET /animal/elephant.gif HTTP/1.1
   Host: www.animalplanet.com
2. The Web server sends a response message to the cache with the requested object.
   HTTP/1.1 200 OK
   Date: Sat, 10 Aug 2020 15:30:30
   Server: Apache/1.3.0 (Unix)
   Last-Modified: Wed, 7 July 2020 10:22:16
   Content-Type: image/gif

   (data ……..)

3. The cache forwards this object to the requested browser and also stores a copy locally along with its modified date.
4. After some time if another browser requests the same object to the cache, it performs an up-to-date check by issuing a conditional GET by sending the following message.
   GET /animal/elephant.gif HTTP/1.1
   Host: www.animalplanet.com
   If-modified-since: Wed, 7 July 2020 10:22:16
   The If-modified-since message asks the server to send the object only if has been modified since the date sent in the request message. If the object has not been modified then the following response message is sent.
   HTTP:/1.1 304 Not Modified
   Date: Sat, 15 Aug 2020 10:25:30
   Server: Apache/1.3.0 (Unix)

   (empty entity body)

The above message does not include the requested object thus reducing the bandwidth wastage and the response time perceived by the user.

3.5 File Transfer Protocol (FTP)

It is a protocol that uses a client-server architecture to transfer files between a local host (client) and a remote host (server) over the Internet.

Fig. 3.11 shows the movement of files between local and remote systems. The user first accesses the remote system by providing his/her identification and a password. Then the files get transferred between the local and remote systems.

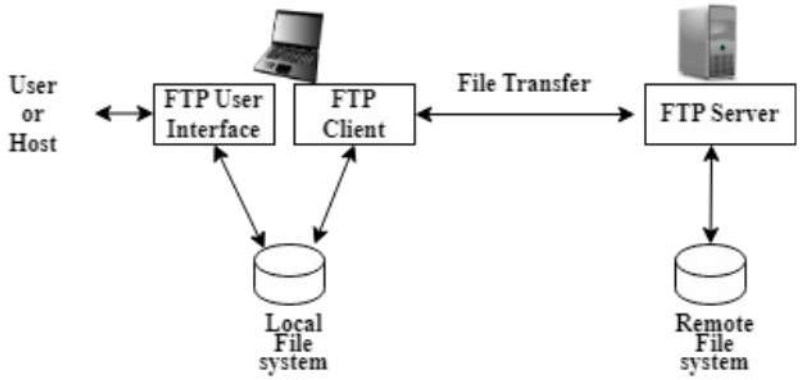

Fig. 3.11: Movement of files between local and remote file systems

FTP uses two parallel TCP channels or connections to transfer files: The control channel and the data channel.

Control channel: Used to send control information such as identification of the user, password, commands to put and get files, and change directories.

Data channel: Used to send the actual data (files).

Steps used to transfer files using FTP:

1. First the client uses port no 21 to initiate a control TCP connection with the server.
2. The client then sends his identification and password over the control connection.
3. A command to change the remote directory is then sent by the client on the same control connection.
4. Upon receiving the file transfer command, the server now initiates a TCP data connection to the server.
5. The server then sends only one file over this connection and closes it.
6. If another file has to be transferred, a new data connection has to be established but the commands can be sent over the same control connection as it remains open throughout a user session.
7. Also, throughout the duration of the user session the server keeps track of the current directory of the user and should maintain the state of the user.

3.5.1 FTP Commands and Replies

FTP commands are sent from a client and the replies are sent back by the server.

Some commonly used FTP commands are:

1. USER <username>: A client sends user identification to a server.
2. PASS <password>: A client sends a user password to a server.
3. LIST: A client requests the server to send a list of all the files in the current remote directory.
4. RETR <filename>: A client requests the server to retrieve a file in the current remote directory.
5. STOR <filename>: A client requests the server to store a file in the current remote directory.

NOTE: For every command given by the user, the FTP protocol sends a command on the control connection.

FTP replies are sent by the server for every command sent by the client. These replies begin with a 3-digit number followed by a phrase in the status line.

Some common reply messages are given below:

1. 425 Cannot open the data connection
2. 331 User name ok but the password is required

3.6 Self-Assessment Questions

Q1. Briefly explain the architectural issues related to network applications (8 marks, L3)

Q2. What are the transport services on the Internet to the applications? (10 marks, L2)

Q3. Write a short note on HTTP protocol (6 marks, L2)

Q4. What is an Internet Cookie? Explain how it is created and used. (10 marks. L3)

Q5. Briefly explain the FTP protocol (6 marks, L2)

Q6. Discuss Web Caching and its benefits. (8 marks, L3)

Q7. Explain the use of conditional GET messages with an example. (8 marks, L3)

Q8. List some popular applications operating at the application layer (4 marks, L2)

3.7 Self-Assessment Activities

A1. Prepare a table of all the popular applications, the transport layer protocols used by them, and the port numbers used for communication on the Internet.

A2. Implement a client-server application that uses any application layer protocol to communicate over the network.

A3. Use any open-source packet capture tool to analyze the network traffic and identify the application layer protocols used by them.

A4. Find out the challenges of providing security at the application layer.

3.8 Multiple-Choice Questions

Q1. Protocol operating at the application layer. [1 mark, L1]

A. UDP
B. ICMP
C. FTP

D. ARP

Q2. Application layer protocol for sending and receiving emails [1 mark, L1]

A. HTTP
B. FTP
C. SMTP
D. None of the above

Q3. Application layer protocol for accessing web pages [1 mark, L1]

A. HTTP
B. FTP
C. SMTP
D. None of the above

Q4. Application layer protocol for transferring files [1 mark, L1]

A. SMTP
B. HTTP
C. FTP
D. None of the above

Q5. The disadvantage of using a client-server network connection is, [1 mark, L1]

A. Cost
B. Security
C. Scalability
D. Performance

Q6. The disadvantage of using peer-to-peer network connection is, [1 mark, L1]

A. Cost
B. Security
C. Scalability
D. Performance

Q7. HTTP method to extract information from the server, [1 mark, L1]

A. GET
B. PUT
C. POST
D. CONNECT

Q8. The time needed to request and receive an HTML file in a Non-persistent connection is, [1 mark, L1]

A. Zero RTT + file transmission time
B. One RTT + file transmission time
C. Two RTTs + file transmission time
D. Three RTTs + file transmission time

Q9. The purpose of an Internet cookie is,

A. Improving computer performance
B Storing files on the user's computer
C. Providing security to user's applications
D. Tracking user interaction and preferences

Q10. An internet cookie is,

A. An image downloaded from the Internet.

B. A computer virus

C. A small text file stored on the user's browser

D. None of the above.

Q11. An internet cookie is transferred between the server and the user's browser through,

A. An image downloaded from the Internet.

B. A computer virus

C. An email

D. HTTP headers

Q12. Web caching is,

A. A technique used for securing web pages

B. A technique used to store copies of web content for reducing load time.

C. Both A and B

D. None of the above.

Q13. A web cache is located on,

A. On the website's server.

B. On the user's computer.

C. On a separate server called as a Proxy server.

D. None of the above.

Q14. Which of these is not an FTP command?

A. USER

B. PASS

C. STOR

D. PATCH

3.9 Keys to Multiple-Choice Questions

Q1. FTP (C)

Q2. SMTP (C)

Q3. HTTP (A)

Q4. FTP (C)

Q5. Cost (A)

Q6. Security (B)

Q7. GET (A)

Q8. Two RTTs + file transmission time (B)

Q9. Tracking user interaction and preferences (D)

Q10. A small text file stored on the user's browser (C)

Q11. HTTP headers (D)

Q12. A technique used to store copies of web content for reducing load time (B)

Q13. On a separate server called a Proxy server (C)

Q14. PATCH (D)

3.10 Summary of the Unit

This unit covers the three important topics to be studied at the application layer: architecture used to build Internet applications, transport layer services available to the applications, and protocols used to control and manage these applications. The first topic discusses the architectural concepts of the application layer and the implementation of network applications. The two important architectures: client-server and peer-to-peer architectures are studied and compared. The application processes running on remote computers communicate through messages and the endpoints through which they communicate are the sockets, which consist of a pair of information: port number and IP address. Sockets are used to connect the application and the transport layer and form the application program interface (API). The IP address gives the destination host address and the port number gives the identification of the process on that host.

The second topic covers the four important services available to applications and their data from the transport layer: reliability, throughput, timing, and security. Reliability is provided through TCP protocol at the transport layer. Throughput and timing depend on bandwidth availability. Security is provided through the enhancement of the transport layer by the addition of the SSL layer.

In the third topic, two important protocols are discussed: HTTP, and FTP. HTTP is used to access web pages, and FTP is used for transferring files between hosts on the Internet. This topic also covers the Cookie technology which is used to track the user's activity on an Internet website and their preferences. To improve the response time for a user, the time to load a requested web page can be reduced. This is made possible by the use of web caching or proxy servers.

3.11 Recommended Learning Resources

[1] James F Kurose and Keith W Ross, Computer Networking, A Top-Down Approach, Sixth Edition, Pearson,2017.

[2] Behrouz A Forouzan, Data and Communications and Networking, Fifth Edition, McGraw Hill, Indian Edition

[3] https://www.tutorialspoint.com/http/http_methods.htm

[4] https://www.simplilearn.com/data-encryption-methods-article

[5] https://www.geeksforgeeks.org/caesar-cipher-in-cryptography/

[6] https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies