Chapter 2: Application Layer

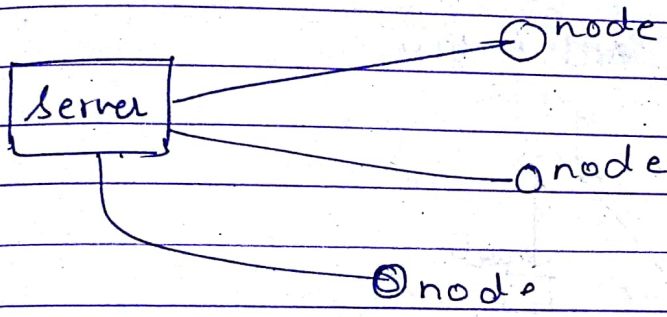
* Principle of Network Application:

→ Application Architecture → Client Server
                          → Peer to peer

→ Processes communicating
→ Sockets
→ Addressing processes
→ Transport services required
→ TCP and UDP

→ Application architecture

• Client Server architecture:



In this model there is a host called 'server' which recieves requests from many other devices called the 'clients'

A classic example for this architecture is web service request. ~~when~~ from browsers on the client. When web server receives request from the browser and responds for the requested object.

The client server architecture cannot communicate between two clients that is two web pages on client cannot communicate directly.

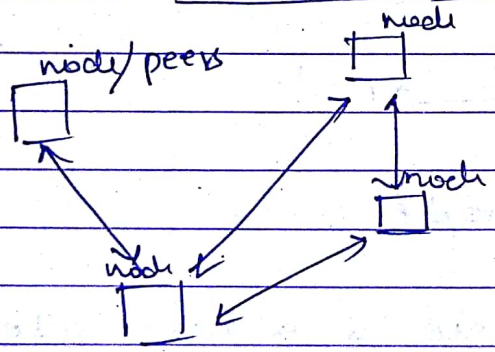
Since the servers remains same all the time it has a fixed IP address. The IP address of the client changes every time it enters the n/w.

Bcoz the server has fixed IP address the client can send its packets to the same IP address (well-known address) and bcoz also bcoz the server is always on

Few ex for this are Web, FTP, Telnet & Email.

Often in this application a single architecture cannot handle all the client requests. ∴ for this reason a 'data center' housing a large no. of hosts within which creates a powerful virtual server; Google, Bing, Gmail, Amazon, ebay employ one or more virtual server.

- Peer to Peer Architecture



There is no dedicated servers in this architecture Instead the application directly communicates between the intermittently connected hosts called peers. The peers are not owned by any service providers but are desktops and laptops controlled by users. Ex: bit torrent, Skype, PPstream etc.

There can be hybrid n/w that is both client server and P2P The client server model can be used to find the IP address of the user and further user to user communication is via P2P

one of the most compelling feature of P2P architecture is their "self scalability" ie in P2P file sharing application although each peer generates workload by requesting files, each peer also adds service capacity to the system.

Since peers are intermittently connected and change IP addres for every new connection hences complexity increases.

*Process Communicating.

Process communicates bet^n the applications A process can be thought of program that is running within an end system. when process are running with in same end system they can communicate with ~~eather~~ each other with interprocess communication

Process on two different end systems communicate with each other by exchanging messages across computer n/w

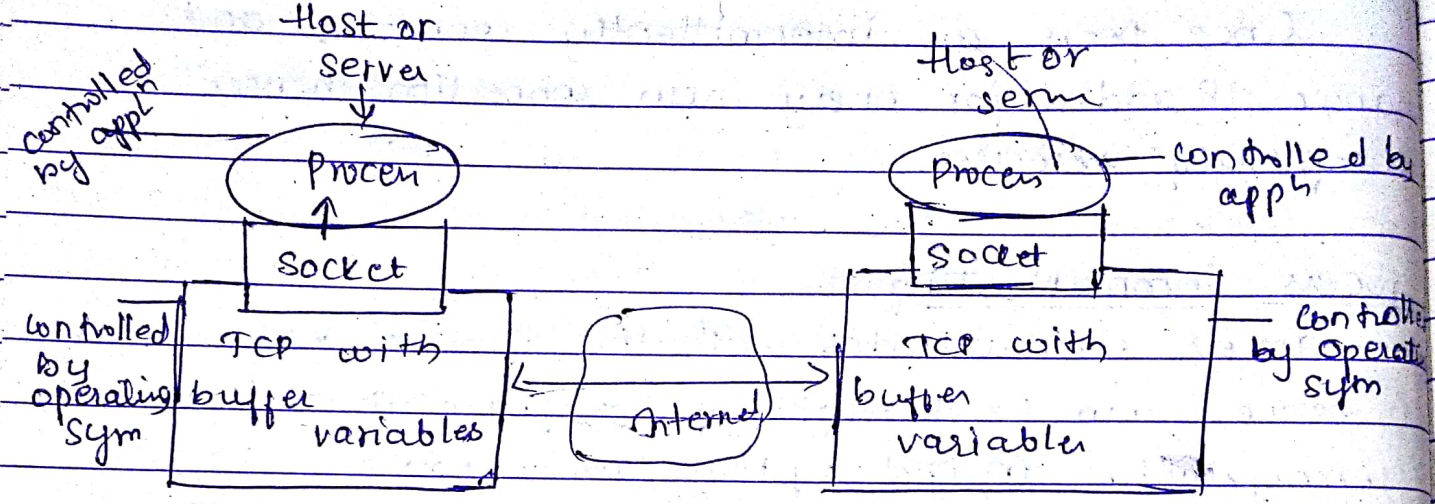
Client and Server Processes.

In the context of a communication session between a pair of processes, the process that initiates the communication is labled as 'Client'. The process that waits to be contacted to begin the session is the server.

{ ^Explain about client-server & Peer-Peer transfer^ }

Socket:

A process sends message into and receives message from the n/w through a software interface called a 'socket'



Addressing Processes.

In the Internet the host is identified by its IP address. It is known that IP addres is a 32 bit quantity that is used for unique identification to identify the host

Identifier includes both IP addres and port nos associated with process on host

ex for port numbers.

HTTP server 80

mail " 25

IP addres 128.119.245.12

port number 80

• Transport Services Available to Applications :

→ Reliable Data Transfer :-

→ Throughput

→ Timing

→ Security

Reliable Data Transfer:

Packet loss can happen if there is overflow in buffer of a router or discarded by the router or lost with in a n/w or the bits are corrupted.

This is undesirable in many applications such as email, FTP, Remote host access, web document transfer. ∴ App provides transfer protocol that helps in acheiving reliable data transfer.

If reliable data transfer is achieved by the n/w then the application can just dump data into the socket and know that there will be reliable data transfer.

Throughput:

Its the rate at which sending proces can deliver bits to the receiving proces.

Because other proces will be sending & receiving context their is bandwidth sharing and throughput can fluctuate. Hence this led to another protocol in application layer that guarantees the throughput. It always ensure that the " is rbits/sec.

Application which require throughput are said to be "bandwidth sensitive Applications".

"Elastic Applications" make use of as much as less throughput that is available.

Timing:

Provides timing guarantee along with throughput.

ex: guratee might be that every bit the sender pump to the socket arrives @ the reciever not more than 100msec later.

Such service would be appealing in real time applications.

For non-real-time applications lower delay is alway preferable to higher delay, but no constraints to end to end delay.

Security:

Provides one or more security services.

ex: @ sending host a protocol can encrypt all the data transmitted and @ the recieving host all the data can be decrypt by the protocol.

Such servica provides confidentiality can Also provide data integrity and end point authentication

TCP services:
→ connection oriented service:-

TCP Services:
- The model includes connection oriented services and reliable data transfer service.
- TCP server & client exchange transport layer control information with each other before message flow starts called as handshaking procedure.
- After handshaking - TCP connection takes place.
- TCP connection is said to be set up bet^n sockets of the 2 procers.
- These are full duplex connection.
- When the job of the application is done, it must tear down the connection
- It also includes congestion control mechanism.

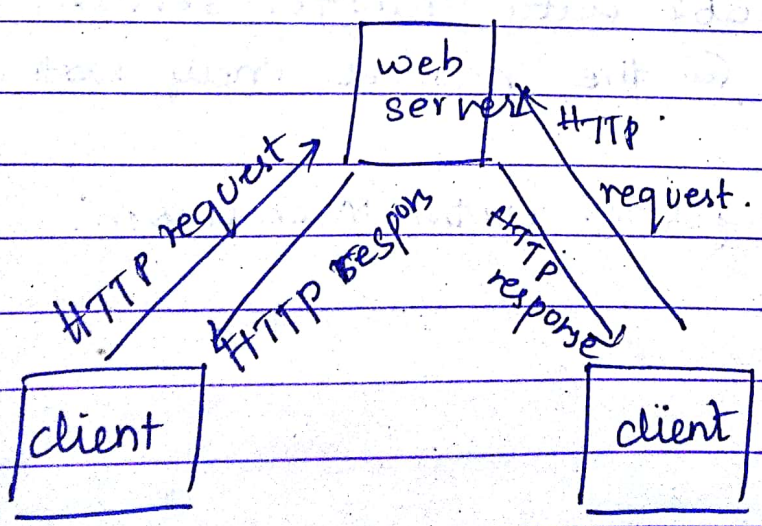
UDP Services
- Its a no-frills, lightweight transport protocol.
- Connection less .: No handshaking
- provides unreliable data transfer service.
- Messages arriving @ the reciever may not be in order.
- doesnot include congestion control mechanism.

HTTP

- Hyper Text transfer protocol.
- Its defined in RFC 1945 and RFC 2616
- Implemented in 2 programs
  - client program
  - server "

→ Both programs execute on diff system talk to each other by exchanging HTTP msg.

- HTTP defines the message structure and how the client & server exchange msg
- HTTP define how web clients request for web pages from web servers and how the server transfers.
- HTTP uses TCP as its underlying transport protocol.
- It first initiates TCP connection with server once connection is established there is accen for the browser & server.
- The client sends HTTP request msg to socket and recieves response ||y with the server. Then the actual transmission is carried out



- The server sends requested file to clients without storing any information about the client.
- ie if a same client request for a file twice it will not respond saying that the file is already served instead sends the file again.
- Since it maintains no information about clients HTTP is a "stateless protocol"

HTTP - Non Persistent connections:

1) HTTP client process initiates a TCP connection to the server. Associated with TCP connection there will socket at the client and at the server

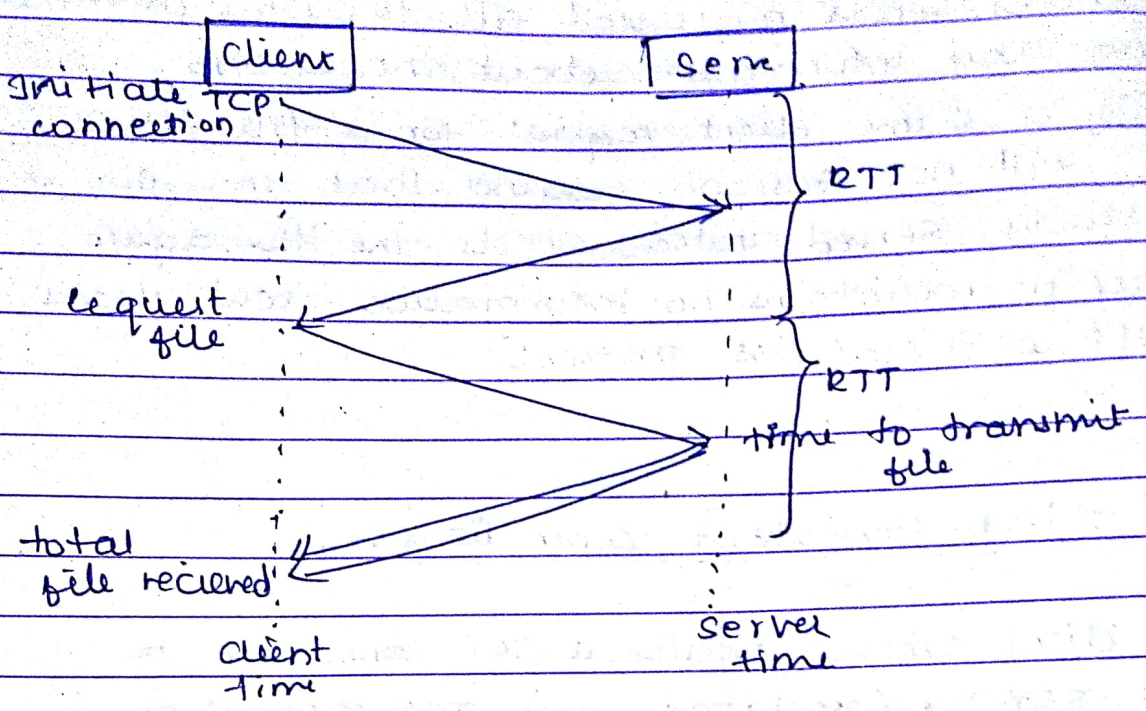
2) HTTP client sends request msg to server via socket.

3) HTTP server recieve the request msg and the service is provided

4) HTTP server tells TCP to close the TCP connection

5) HTTP client recieves the response msg and terminates the TCP connection

6) The first 4 steps is repeated for the next request.



Non persient http response time:

RTT: time for small packet to travel from client to server & return back.

- one RTT to initiate TCP connection
- one RTT to request for HTTP
- Response time = 2 RTT + transmission time.

issues

- Require 2RTT per object
- OS overhead for each TCP connection
- browsers often open parallel to fetch referenced objects.

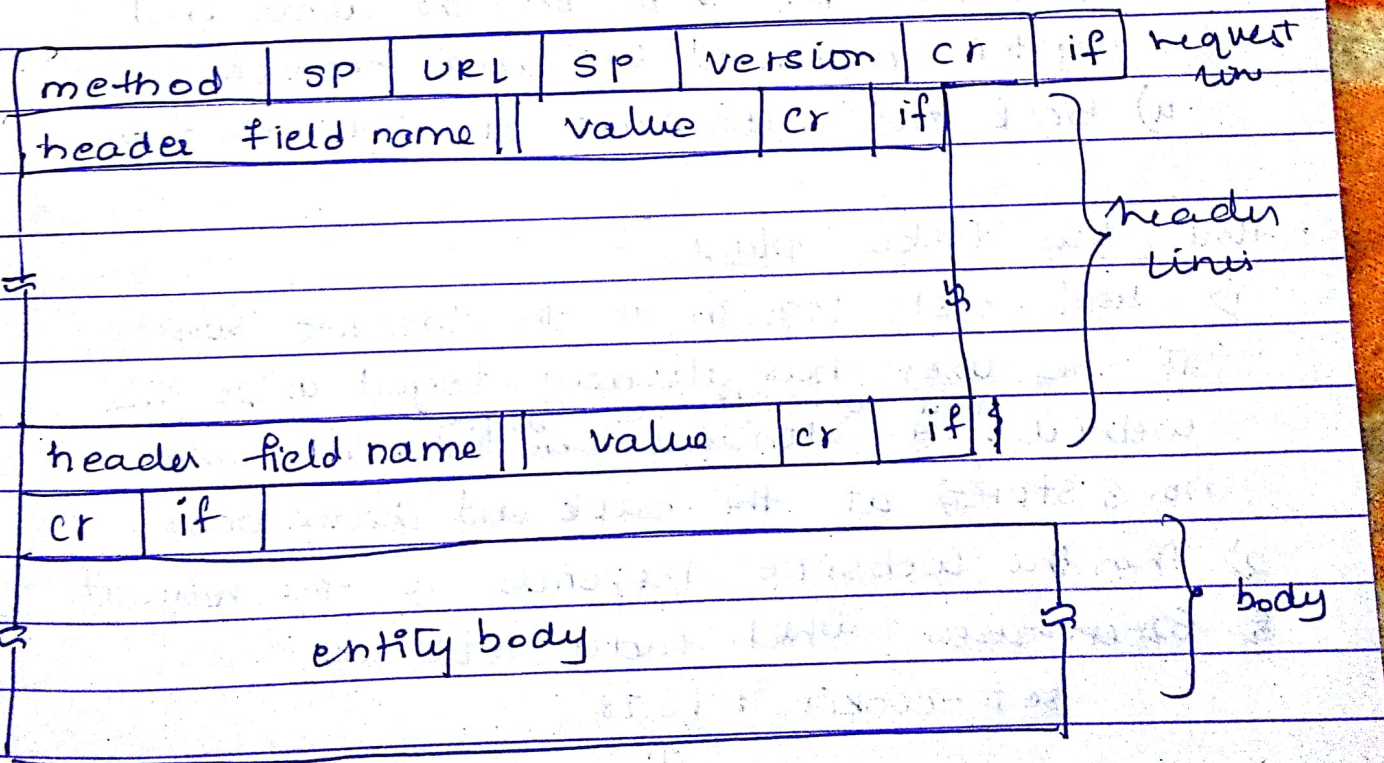
Persistent HTTP

- Here the server leaves TCP connection open after sending a response.
- Subsequest response & request bet^{n} same client and server can continue over that same connection.
- In particular, an entire web page can be sent over TCP connection

ex - suppose if 10 images are requested then the request are made back to back without waiting for the reply.

- but the connection is terminated after certain idle time.
- default HTTP uses persistent connection.

HTTP request msg format:



HTTP response status code

→ status code appears in 1st line in server to client response msg

↔ ex:

- 200 OK
- 301 moved Permanently
- 400 Bad Request
- 404 Not found
- 505 HTTP version not supported.

Cookies:

This technology allows sites to keep track of user

This has 4 components

1) a cookie header line in HTTP response msg
2) " " " " " " request "
3) a cookie file kept on the user's end system managed by user's browser
4) back end data base at the web site.

How this takes place

1) Client host logs in to the Internet Suppose if the user has already loged in to the web site, the website assign unique identification no. & stored at the back end data base.
2) Then the website responds to the request.
3) which may look like

set-cookie : 1678

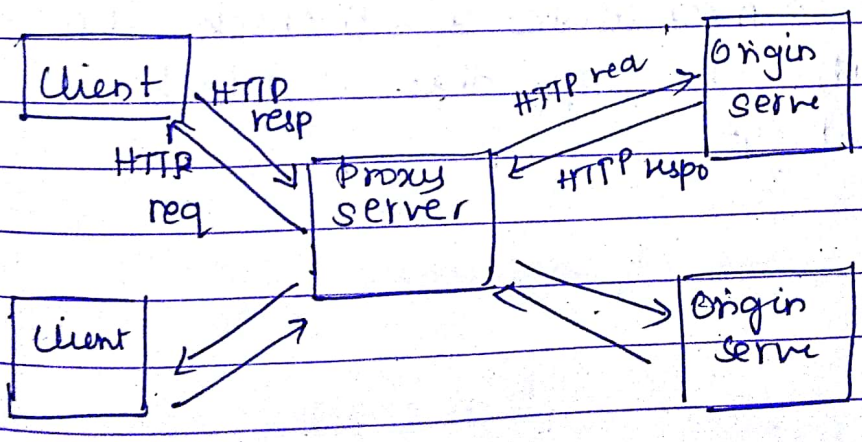
↑

unique id.

→ User redeve the HTTP response message. The browser then appends a line to the spl cookie file that it manages.
→ every time user browses the web site the cookie header is stored ~~in~~ it thus the web servers keep the user track.
→ ~~If that user~~ Though the website doesn't know the details of the user but it will have the track that the user with Id 1678 logged in to its website and what is the track and ~~how~~ for how long the site is being used.
- If the user links all his information on the site then the database may store all the data along with the unique Id.

Web Caching

web cache also called a proxy server.
its a n/w entity that ~~satisfy~~ satisfy the HTTP request on behalf of the server.
- It has its own disk storage and keep the resent requests.



→ TCP established connection to Proxy server or web cache and sends HTTP request
→ Web cache checks to see if it has the copy of the object stored. If it does it returns the object with HTTP response msg to client
→ If it doesn't have the object. It makes TCP connection to origin server and sends HTTP request, after receiving the request the origin serves response to webcache along with the object
→ The web cache recives the object and stores a copy on it and sends it to client with a response msg.
– Reduces system response.

The Conditional GET

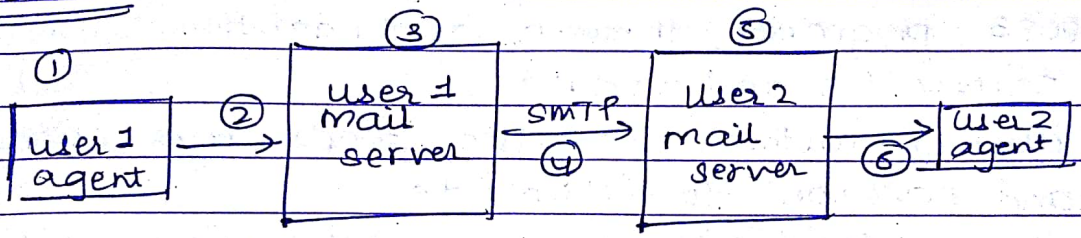
– Web cache introduce a problem that it keeps a copy of object it that object is modified in the origin server the copy in the web cache is stale.
HTTP has a mechanism which checks this and updates the copy this is called conditional GET

Sunidhi

Mail Access Protocol
- SMTP
- IMAP
- POP3

* SMTP - Simple Mail Transfer Protocol
- defined in RFC 5321
- transfers msg from sender's mail servers to the recipients mail servers.

Operation:



① user 1 invokes their user agent for e-mail provides user 2's email address. composes msg and instructs to send msg
② user 1's agent sends msg to the user 1 mail server where it is placed in queue
③ The SMTP running on client side server at ~~see~~ user 1 sees the message in ques. opens TCP connection to SMTP server running on user 2 server
④ After initial SMTP handshaking. SMTP client sends user 1's message to user 2
⑤ At user 2's mail server SMTP recieves the message and places in User 2's mail box
⑥ User 2 invokes his user agent and read the message at his convience.

Mail Access Protocol

1) SMTP
2) POP3
3) IMAP

1) SMTP

POP3 Post office Protocol

- defined in RFC 1939
- short and reliable and simple.
- begins when an user agent opens a TCP connection
- POP3 progresses through authorization, transcation and update.
- during authorization user sends user name and password to authenticate
- During transaction the user retrives msg also user can perform other operations
- update occurs after the client has issued the quit command.
- In transcation phase the user issues commands and the server responds to each command with reply the possible replies are +OK for correctness and -ERR for wrong indication
- download and delete mode
  where msg cannot be read after the client is changed
- download & keep modes.
  msg can be retrived later.

IMAP - Internet Mail Access Protocol.
defined in RFC1939
- more complex compared to POP3.
- The recieved msgs are always stored in the associated folder
- The uses can perfor different action of the msgs.
- This protocol allows users to make folders and sort the msg in diff folder.
- Allows user to search remote folder that has some matching criteria.
- Serves and maintain user state & information of IMAP sessions