1

Omkar Masur
180905330
Section D
44

Q2] Sender host transmits first packet to switch, the transmission time is $5000/10^7$ which is 500 μs. After 500 μs, second packet is transmitted. First packet reaches destination in $500 + 35 + 20 + 20 + 500$ $= 1075$ μs. While the first packet is travelling to destination, second packet starts its journey after 500 μs and rest of the time taken by second packet overlaps with first. So overall time is $1075 + 500 = \underline{1575}$ μs

Q1]

| Circuit Switching | Packet Switching |
| --- | --- |
| 1) Each data unit knows the entire path address which is provided by source | 1) Each data unit just knows the final destination address, intermediate path is decided by routers |
| 2) Data is processed at source system only | 2) Data is processed at all intermediate node including source system |
| 3) Delay b/w data units is uniform | 3) Delay is not uniform |
| 4) Not a store and forward technique | 4) It is a store & forward technique |
| 5) Inflexible in nature since data packets are routed along same dedicated path | 5) Flexible, where the different data packets follow different paths. |

2

| | |
|---|---|
| c) Entire message is recieived in the order sent by source | c) Individual packets of the message may be received out of order. |

Distance = 50 km = $50\times10^3$ m

Propagation delay = $50\times10^3 / 2\times10^8 = 25\times10^{-5}$ sec

Bandwidth = size / transmission delay

$\therefore$
$$= \frac{100\times8}{25\times10^{-5}}$$
$$= 32\times10^5 \text{ bits/sec}$$
=

For 512 byte packet,

$$\text{Bandwidth} = \frac{512\times8}{25\times10^{-5}}$$
$$= 163.84\times10^5 \text{ bits/sec}$$

③

Q3] i) TCP socket address has two components, the IP address and the ~~so~~ port number. IP address of a host is nothing but the identifier of a host in a network. Port number belongs to a process withing a host. It is used to uniquely identify the process ~~with~~ or application with a host. eg:- Port 80 belongs to HTTP.

With the UDP server, there is no welcoming socket, and all data from different clients enters the server through one socket. With TCP, where is a welcoming socket and each time a client initiates a connection ~~to~~ a server, a new socket is created. Thus, to support n simultaneous connections server would need n+1 sockets.

Q4]

(ii) Since the application ~~day~~ needs the responsibility of delivering the bytes in order to the other side, we will need to use TCP protocol here. This is because TCP protocol is the protocol which does just that.
TCP creates a stream and accepts the responsibility of delivering bytes in order to the other side.
~~This~~ It is a full-duplex communication.

(iii) Since occasional loss of messages is acceptable, UDP protocol can be used here. ~~...~~ UDP is a connection-less protocol which takes no responsibility of delivering the application layer messages.

(i) UDP can be used because each datagram can be used for each chunk of data.

8

Q5] a) GobackN:

Host A sends segments 1, 2, 3, 4, 5, 6 initially, and the later resends 3, 4, 5, 6. Hence, A sends 10 in total. B sends ACKS for 1, 2, 4, 5, 6 initially and then resends ACKs for 3, 4, 5, 6. Hence B sends 9 ACKs in total. They are 3 ACKS with sequence no. 3, 2 with 1, 2 [initially] & then to 4 remaining with 3, 4, 5, 6

Selective repeat:

A sends 7 segments in total, initial 1, 2, 3, 4, 5, 6 and then 3 again. B sends 6 ACKs. They're 5 ACKs with seq. no. 1, 2, 4, 5, 6 and the one ACK with seq. no. 3.

TCP:

A sends 7 segments in total, initial 1, 2, 3, 4, 5, 6 and then 3 again. B sends 6 ACKs.
B sends ~~1~~ ACKs with 2, 4 with seq. no 3 and then with seq. no. 7

b) TCP, because TCP uses fast retransmit without waiting until time out.

(C)

Q6] 1) We use, we use HTTP cookies. Cookies allow sites to keep a track of users. Most commercial websites use it today.

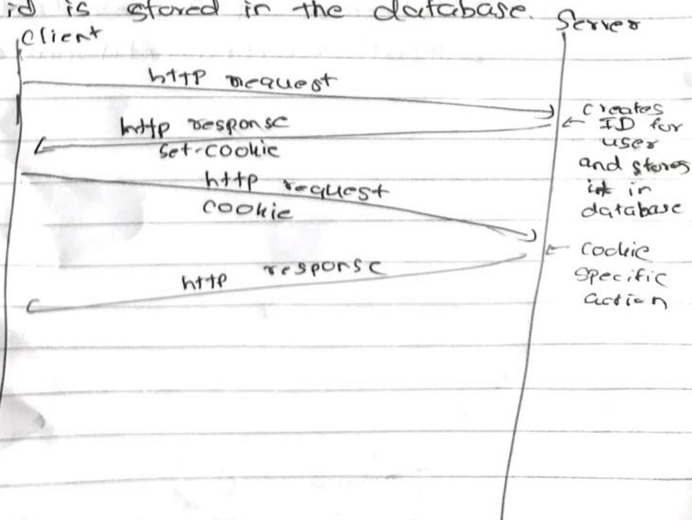
2) Cookies have four components: header line in HTTP response, cookie header line in HTTP request, cookie file kept in the user's end system, a backend database at the website.

3) When a user contacts a website for the first time, the server creates a unique identification number and creates an entry in its backend database, which is indexed by this id.

4) The server then responds to the client's browser, including in the HTTP response Set-cookie: header, which has the id.

5) As the client continues to browse the website each time he requests a web page, the browser consults the cookie file, extracts the id and puts the cookie header line which icludes the id in the request.

6) The backend server then is able to identify this client based on the id, since this id is stored in the database.



A brand new connection must be established and maintained for each new requested object in HTTP ~~[illegible]~~ 1.0. For each of these connections TCP buffers must be allocated and TCP variables must be kept in both client & server. This can place a burden on the webservers, which may be ~~[illegible]~~ serving requests from hundreds of different clients simultaneously. ~~B~~ Also, ~~[illegible]~~ each object suffers a delay of 2 * RTTs. With HTTP 1.1, the server leaves the TCP connection open after sending a response. Subsequent requests & responses b/w same client & server can be sent over the same connection. In particular, an entire web page including the images can be sent over a single TCP connection, hence reducing the load on the webserver.

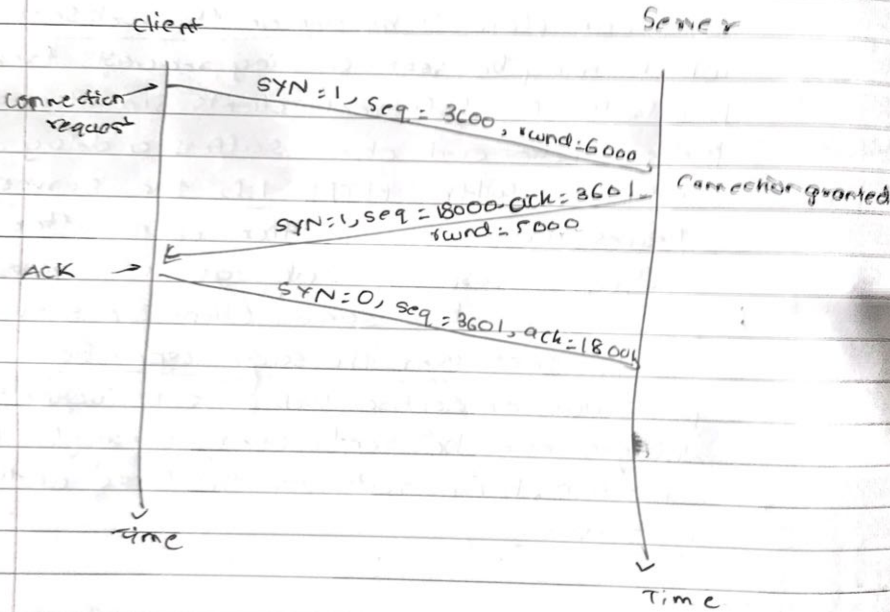
Q7]

$X = (44 + 100) * 25$

$= 3600$

$Y = 3600 * 5$

$= 18000$




1) Client side TCP first sends a special TCP segment to Server TCP. It has no application layer data. But one of the flags, SYN is set to 1. Client randomly chooses seq as 3600, and puts it in the TCP segment

2) Once IP datagram containing TCP SYN arrives at Server, server extracts the TCP SYN, allocates TCP buffers and variables to the connection and sends a connection granted segment to the client. It also has no application layer data. ACK is set to 18000 randomly.

3) Upon recieving SYNACK segment, client allocates buffers and variables to the connection
Client host the sends Server another segment
this last segment acknowledges the server's connection granted segment. Ack is set to seq+1, where seq was ~~recieved the~~ recieved from server previously.