ASSIGNMENT - 01 MID - SUMMER 2022 SET B

NAME: NSHIMYUMUKIZA THIERRY

ID : 23301701

SEC : 22

1. a) I disagree with the statement;

Cookies are stored locally on the user's device, not on the browser server. when you visit a website, the server sends small cookies files that your browser saves. These cookies help remember preferences, sessions or login states.

If you format your computer, all stored cookies are erased the sever may still have some data (like account information) but it can not restore cookies unless you log in and recreate them.

therefore, the sever does not store cookies it only reads or creates them when you visit.

b) i) No, the new peer cannot finish the download if only two peers are ON.

- Reason: 1.5 GB = 1.5x1024 = 1536MB → 1536 Chunks.
  two peers provide 2x512 = 1024 distinct chunks which is less than 1536.

ii) It is possible.

because . youtube uses adaptive bitrate streaming
. the player might have enough buffered data
. local / network caching can cover short term needs

c) i) first access, 16 July 2022 at 13:01:22
second access, 19 July 2022 at 11:01:23
time difference:
= 2 days 22 hours and 1 sec
= 2 x (24) + 22 = 70 hours

TTL = 48hrs

70 - 48 = 22hrs after expiration.
therefore, DNS record must be fetched again

therefore, ~~Better~~ 4 x 55 = 220ms
DNS RTT = 220ms

ii) 1 RTT for TCP

1 RTT for Base HTML

20 RTT for 20 Additional objects

total RTT known = 22 RTTs

RTT = 2(35)

= 70ms

so, total RTT = 22 × 70

= 1540 ms

iii)
- DNS RTT = 220 ms
- RTT object = 1540 ms
- download time = 125 ms
- no of object = 21

  Download objects = 21 × 125 = 2625

total_time_to load_webpage = DNS RTTs + (22 RTTs × 70) + (21 × 125)

= 220 + 1540 + 2625

= 4385 ms

---

(2) a). Source port : a dynamic / ephemeral port (eg: from range 1024 - 65535) chosen by outlook for SMTP

- Destination port: 25, 587 (SMTP) for sending mail from Rafiq to his SMTP server, then the final hop to yahoo uses SMTP 25 between mail servers.

b) i) ∘ Data bytes sent not acknowledged = S_f = 100 to $S_n - 1$ = 150

51 bytes sent and waiting Ack

∘ window size = 100 bytes, so usable window = 100 - 51 = 41 bytes can be sent now.

ii) If an Ack is lost, the sender uses a timeout mechanism. When timeout occurs, it retransmits the unacknowledged data. the receiver already received that data, so it discard duplicate byte resend the Ack.

(2) c) i) - Server ISN = 5549

- HTTP request 1 [3rd segment of 3-way handshake] had length = 569 bytes
- Client ISN = 9666

  Seq = 9666 + 1 = 9667

so, - SYN for client seq = 9666

- SYN-Ack seq = 5549, Ack = 9667
- Ack + HTTP then

  Seq = 9667 + 569 = 10236

therefore, Datasent for server

Seq = 5549 + 1 = 5550

Ack = 10236

ii) HTTP request 1 = 569 bytes

HTTP request 2 = 999 bytes

total = 569 + 999 = 1568 byte

Initial sequence = 9666 + 1 = 9667

Next byte to be sent = 9667 + 1568

= 11235

Rwnd = 8000

(3) a) i) The correct order (OSI model at Bob's side);

1. The data is transmitted over the medium and received (physical)
2. Identifies ~~that this is~~ in the correct hope address (Network)
3. Identify that this is indeed correct host (Network)
4. Identifies the process of delivering the message - (Transport)
5. Controls sessions - (Session)
6. Decrypt data (presentation)
7. Bob reads the message - (Application)

ii) Mac address (layer 2 addrs) changes at each hop.

b) i) → TTL is incremented each time to make the packet reach one hope further - when TTL = 1, the router decrements it to 0, discards packet, and sends ICMP Time Exceeded back. originator ~~sends~~ learns first router's address.

→ TTL = 2 reaches second router, then ICMP back, originator knows destination reached when it receives ICMP ~~echo~~ Reply (not time Exceeded) from destination.

c) i) Number of fragment

$= \frac{21739}{1992} = \lceil 10.917 \rceil$

$= 11$ packets

MTU = 2038 bytes
Header = 46 bytes
Max data per fragment = 2038 - 48 = 1992
total = 21739

ii) MF of last packet = 0 (last fragment)

iii) - last packet data size = 21739 - (10 × 1992) = 21739 - 19920 = 1819 bytes
- last packet total size = 1819 + 46 = 1865 bytes

(iv) offset = data bytes before the fragment / 8

offset for 9th fragment = 9 × 1992/8 = 17928/8 = 2241