

山 1973. 1. 10. 10:00-11:00 100m. 111

(a) Hence is the correct order according to

(1) simple multilayered  $\leftarrow$  (2) simple multilayered  
the OSI model:

$$6 \rightarrow 5 \rightarrow 3 \rightarrow 15 \rightarrow 2 \rightarrow 7 \rightarrow 4$$

1. Application Layer (6)
  2. Presentation Layer (5)
  3. Session Layer (3)
  4. ~~Presentation Layer (5)~~ Transport (1)
  5. Network Layer (2)
  6. Data Link Layer (7)
  7. Physical Layer (4)

(b)  Yes, it is possible for a new client to download the torrent successfully with 10 peers left in the swarm. In the BitTorrent protocol, "peers" are clients that already possess all or part of the file in the torrent and are currently sharing those parts to other clients in the swarm. The number 10 is sufficient provided the necessary data among those peers.

(c) No, the proxy server will not send a conditional GET request to the origin server if cached copy is still fresh according to its cache headers. A conditional GET is only sent when the cached content is stale or its freshness needs revalidation. Since the proxy already has a valid copy, it directly serves the web page to the client without contacting the origin server.

(d) It's possible due to adaptive bitrate streaming and advanced video compression like H.265 or AV1, which reduce the data size without lowering resolution. These

technologies let you watch 8k video smoothly even with less than 100 mbps bandwidth.

[2]

a) websites track users using cookies, browser fingerprinting, cache identifies and tracking pixels even if users don't sign up.

Tracking helps HTTP by maintaining session states, enabling caching, personalizing content and improving network efficiency

(b)

Recursive DNS lookup is better overall as it reduces client-side load and latency. The resolver handles the full query process and caches results, providing faster and more efficient name resolution for users.

(c)

(i)

$$RTT = 2 \times 23 \text{ ms} = 46 \text{ ms}$$

$$\text{Num of RTT} = 4$$

$$RTT_{DNS} = 4 \times \text{RTT per link}$$

$$RTT_{DNS} = 4 \times (2 \times 23) = 184 \text{ ms}$$

$$(ii) RTT = RTT_{DNS} + RTT_{HTTP}$$

$$\text{object size} = 12 \times 8 \text{ MB} = 96$$

$$RTT_{DNS} = 184$$

$$RTT_{HTTP} = 30 \times 35$$

$$\text{Num of obj} = 30$$

$$= 2693 \text{ ms}$$

$$= 1170 \text{ ms}$$

$$\text{Reg RTT} = 39$$

$$\begin{aligned} \text{Total RTT} &= 184 + 39 + 117 \\ &= 430 \text{ ms} \end{aligned}$$

total transfer RTT = RTT + data transfer

$$= 2602 \text{ ms}$$

(ii)

$$\text{file transfer time} = 2875.25 \text{ ms}$$

(iii) file size =  $30 \times 12 = 360$

$$\text{HTTP Response} + = 5344$$

$$\text{Data trans time} = 5344 - 2602 = 2742$$

$$\text{bits } 360 \times 8 = 2880$$

$$X = \frac{\text{total bits}}{\text{transfer time}} = \frac{2880}{2.742} \\ = 1050.3 \text{ Mbps.}$$

[3]

(a) Yes, a web server can receive multiple HTTP requests on the same port. The server differentiates them using the five tuple, for each connection, while the destination IP and port are the same, each client has a unique source IP and ephemeral port. The server's OS uses this five-tuple to route each request to the correct connection.

(b) Missing fields:

- ① Sequence Number
- ② Acknowledgment Num
- ③ Window size
- ④ Urgent pointer.

Reliability: handled at the application layer using sequence num, acknowledgments, and retransmission mechanisms.

(c)

(i) server ISN = 203, client ISN = 8924

Data segment = 889 B, HTTP = 234 B

4th data seg

$$\text{seq} = 204 + 3 \times 889 = 2871$$

$$\text{ACK} = 8925 + 4 \times 234 = 9861$$

(ii) 10th segment lost

13th HTTP request

$$\text{seq} = 8925 + 12 \times 234 = 11733$$

$$\text{ACK} = 204 + 8 \times 889 = 8205$$

(iii) Client RWND: After 13th data seg, 5 pro

$$\text{Buffered} = (12 \times 889) - (5 \times 889) = 6223 \text{ B}$$

$$\text{RWND} = 10000 - 6223 = 3777 \text{ B}$$

