1.

(a) False. Packet switching may not be faster than circuit switching by the fact that delay between data units in packet switching is not uniform. When there are many packets waiting to be transmitted, the delay can be long using packet switching.

(b) False. : Circuit-switched networks also need to send data over a link, which can produce propagation delay.

(c) False. Although HTTP/1.1 uses a persistent TCP connection, each HTTP/1.1 request is treated independently of previous requests and an HTTP/1.1 server does not maintain any history of these HTTP/1.1 requests or replies.

(d) False. In SMTP protocol, the sender and recipient email addresses only need to be included in header fields (in the email text before the “DATA” command), but not in the email message body.

(e) False. Since checksum is calculated by the addition of different segment contents, if the value of one data item is increased and the value of another one is decreased the same amount, the sum and the receiver cannot detect these changes. Therefore, errors can not be detected in this scenario.

2.

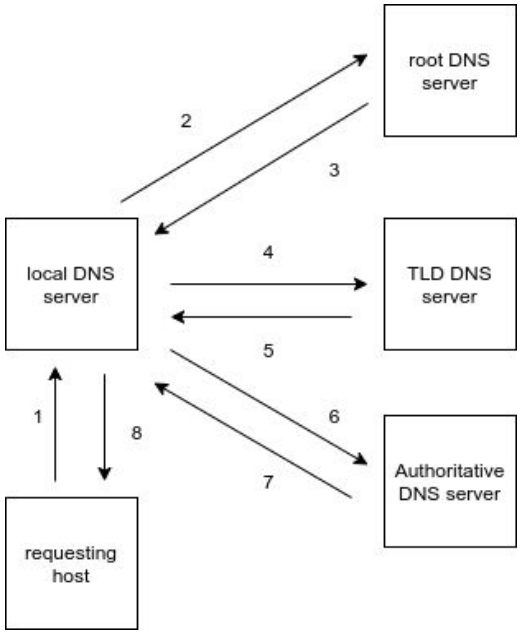
(a) If there is only one single physical root server placed at one location in the world, DNS queries sent from somewhere far away from the root server may experience significant propagation delay. In addition, if there is only one logical root server rather than thirteen, each physical server will need to maintain much more DNS records, which may increase maintenance costs and take the server a longer time to do DNS lookups.

(b) One advantage of using the recursive DNS query at the local DNS server is that it can provide a more accurate or definitive answer since the DNS query is sent recursively between the subsequent DNS servers until the answer is found. One disadvantage of using the recursive DNS query is that it can be quite slow because distances between subsequent name servers can be far away. And if many users are requesting DNS services, the root name server will have a heavy load and users will suffer from delays.

(c) 2ms. If the answer I need is cached in the local DNS server, the time will be minimal. It would take 1ms to send DNS query from to the local DNS server and another 1ms for sending back DNS response to my laptop.

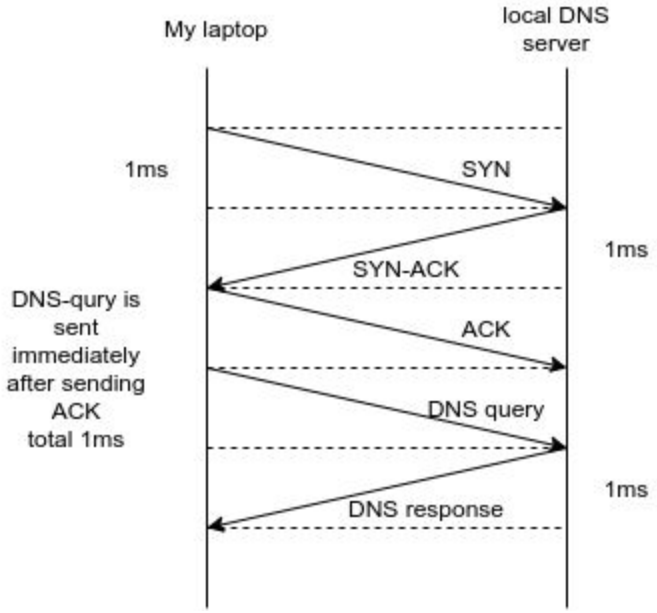
(d) 72ms.
The process is shown in the figure below (steps 1-8)
Round trip from the local host to the local DNS server: 2ms
Round trip from the local DNS server to the root DNS server: 20ms
Round trip from the local DNS server to the TLD DNS server: 10ms

Round trip from the local DNS server to the authoritative DNS server: 40ms
Total: 2 + 20 + 10 + 40 = 72ms



(e)

i.) If TCP is used, my laptop will need to set up TCP connection with the local DNS server first. Assume that my laptop will send the DNS query immediately after sending ACK to the local DNS server for establishing TCP connection, the minimal time it takes my laptop to receive DNS response if the answer is cached at the local DNS server is 4ms.



ii.) Extra time will be needed for establishing TCP connection between DNS servers, the connection process is similar to question i.).

Time for the local host to receive a message from the local DNS server using TCP: 4ms
Time for the local DNS server to receive a message from the root DNS server using TCP: 40ms
Time for the local DNS server to receive a message from the TLD DNS server using TCP: 20ms
Time for the local DNS server to receive a message from the authoritative DNS server using TCP: 80ms
Total: 4 + 40 + 20 + 80 = 144ms
Therefore, the minimal time will be 144ms.

3.

(a) The client will set up a timer after sending the first SYN message. If the client does not receive SYN-ACK before the timer goes off, the client will retransmit the SYN message to the server after the timer goes off. On the server side, since it never received any message from the client, it would do nothing.

(b) The client will set up a timer after sending the first SYN message. If the client does not receive SYN-ACK before the timer goes off, the client will retransmit the SYN message to the server after the timer goes off.
On the server side, what the server will do next depends on its implementation. In some implementations, the server will do nothing until a new SYN is sent from the client. However, in the “classical” implementation, the server will set a timer after sending the SYN-ACK message and wait for ACK from the client. If the server does not receive the ACK message before the timer goes off, it will resend the SYN-ACK message.

(c) Yes, this scenario may happen. For instance, an ACK message sent by the client may somehow stay in the network for some time. This would not affect the normal TCP connection since both SYN-ACK and ACK can be retransmitted. But after the TCP connection is closed, this ACK message can show up at the server side.
The server will simply discard this ACK message since it does not expect this message. The server may only expect an ACK message if it is in “half-open” state. After the previous connection is closed, the server may only deal with new SYN messages.

(d) The main reason for employing the timed wait mechanism is that the client has to make sure the last ACK message can be received by the server. Because the last ACK message can be lost, the server will resend FIN message to the client if it times out, and then the client will resend ACK. If the client closes the connection immediately after sending the last ACK instead of entering into TIME_WAIT state, the server can not normally close the connection once this ACK is lost.

4.

(a)

i.) After reception of these packets, only sequence no.3 is cumulatively acknowledged by the receiver and acknowledgement with sequence no.3 will be sent to the sender.

Because the receiver is expecting a packet with sequence no.4 but it is lost, the window state at the receiver's side will still be 4, 5, 6, 7, 0, 1, 2.
ii.) Since ACK with sequence no.3 is lost, the sender will resend all the packets including sequence no. 3, 4, 5, 6, 7, 0, 1 when its timer goes off. The window state at the sender's side will remain unchanged.

(b)

i.) First, we need to calculate the time to put a single packet onto the link:

$$t_{trans} = \frac{L}{R} = \frac{1\ KB/pkt}{10\ Mbps} = \frac{10^3 \times 8\ bit/pkt}{10^7 bit/s} = 8 \times 10^{-4} s/pkt$$

Round trip time(RTT) for transmitting data would be:

$$RTT = 2 \times 5.6ms = 1.12 \times 10^{-2} s$$

Since the window size is 7, which 7 packets will be transmitted over the link each time, the maximum utilization of the link will be:

$$U_{sender} = \frac{7 \times t_{trans}}{RTT + t_{trans}} = \frac{7 \times 8 \times 10^{-4} s/pkt}{1.12 \times 10^{-2} s + 8 \times 10^{-4} s/pkt} = 46.67\%$$

ii.) Assume when the window size is x, we can fully utilize the link, we have the follow equation:

$$\frac{x \times 8 \times 10^{-4} s/pkt}{1.12 \times 10^{-2} s + 8 \times 10^{-4} s/pkt} = 100\%$$

Resolve the equation, we have x = 15, which means that the window size should be 15 If we want to fully utilize the link.

Since 2^4 = 16 and we can use 4bits to represent 15 numbers, the minimum number of bits we need to represent the sequence number is 4.