Network Systems (201600146/201600197), Test 2

September 14, 2020, 18:15–20:15

(Taken online on campus)

---

1. Switching

1 pt **Q 1.1**

What can be said about the number of distinct VCIs required in a virtual circuit network with **N** nodes, in which up to **m** VCs are used at the same time, in which a switch has at most **n** neighbours, and in which the busiest link carries at most **k** VCs at any time.

A. at least N distinct VCIs are required
B. at least N(N-1) distinct VCIs are required
C. at least m distinct VCIs are required
D. at least n distinct VCIs are required
E. at least k distinct VCIs are required
F. at least k*n distinct VCIs are required
G. at least m/n distinct VCIs are required

→ E

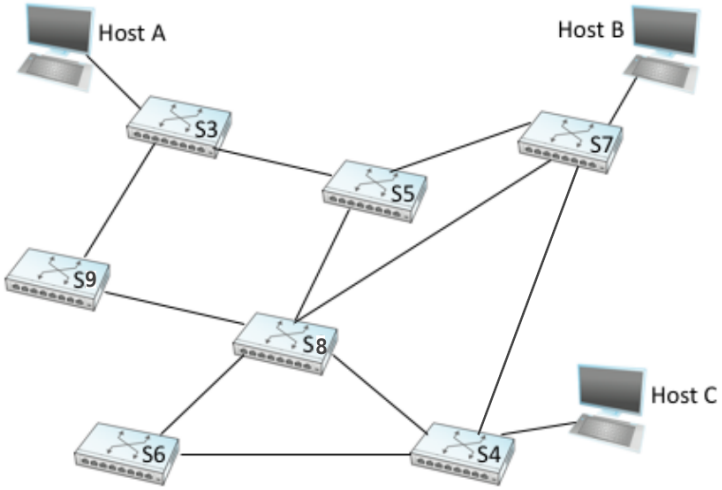
2 pt **Q 1.2** For the same virtual circuit network, what can be said about the number of entries in the forwarding table in an arbitrary switch? Mark all that apply.

A. the forwarding table cannot contain more than N entries
B. the forwarding table cannot contain more than N(N-1) entries
C. the forwarding table cannot contain more than m entries
D. the forwarding table cannot contain more than n entries
E. the forwarding table cannot contain more than k entries
F. the forwarding table cannot contain more than k*n entries
G. the forwarding table cannot contain more than m/n entries
H. non of the above

→ C,F

The total number of entries in one switch cannot be more than the total number of VCs, hence answer C. Furthermore, as each link has not more than k VCs, and and each switch has not more than n neighbours, that switch will not need more than k·n entries, hence answer F.

Please consider the network of switches below.



1 pt **Q 1.3** After running the spanning tree algorithm in this network, which links are not used anymore

for forwarding packets on?

A. S3 - S5
B. S3 - S9
C. S4 - S6
D. S4 - S7
E. S4 - S8
F. S5 - S7
G. S5 - S8
H. S6 - S8
I. S7 - S8
J. S8 - S9

→ C,E,I,J

S3 has the lowest ID, so it will become the root of the tree.

1 pt **Q 1.4** Consider again the network above (before switch S1 was connected to it). Suppose after initialisation and after running the spanning tree protocol, Host A is sending a packet to Host C. As a result, which of the switches will have an entry for Host A in their forwarding table?

A. S3
B. S4
C. S5
D. S6
E. S7
F. S8
G. S9
H. none of the switches

→ A,B,C,D,E,F,G

At this stage, none of the switches know where host C is, so the packet from host A will be flooded over the network, and all switches will learn a path to host A.

1 pt **Q 1.5** After Host C will send a packet back to Host A, which of the switches will have an entry for Host C in their forwarding table?

A. S3
B. S4
C. S5
D. S6
E. S7
F. S8
G. S9
H. none of the switches

→ A,B,C,E

Due to the previous packet, the switches know where host A is, so the reply packet will only take the path from C to A along the spanning tree. Thus, only switches along this path will learn a path to host C.

Suppose now the manager of the previous network connects a new switch, with identity "S1" to the switch S9. The network manager will not do anything to switches S3 - S9 (reconfigure, reboot, ..). What will happen to this network? Give your answer by again indicating which links will not be used for forwarding packets on.

1 pt **Q 1.6**

A. S1 - S9
B. S3 - S5

C. S3 - S9
D. S4 - S6
E. S4 - S7
F. S4 - S8
G. S5 - S7
H. S5 - S8
I. S6 - S8
J. S7 - S8
K. S8 - S9

→ D,E,G,H

A new spanning tree will be made with S1 as its root.

---

2. IPv4

Suppose a router is applying longest prefix matching and has the following table:

Prefix / Length	Next Hop
130.89.130.96/28	Interface 0
130.89.130.103/32	Interface 1
130.89.130.96/27	R2
130.89.130.0/25	R3
129.98.96.0/22	R4
0.0.0.0/0	R5

0.67 pt **Q 2.1** For each of the following addresses indicate to which Next Hop a packet with this address will be forwarded, or select 'drop' if the packet should be dropped instead of forwarded.

130.89.130.104

A. Interface 0
B. Interface 1
C. R2
D. R3
E. R4
F. R5
G. drop

→ A

0.67 pt **Q 2.2** 130.89.130.130

A. Interface 0
B. Interface 1
C. R2
D. R3
E. R4
F. R5
G. drop

→ F

0.67 pt **Q 2.3** 129.98.98.129

A. Interface 0
B. Interface 1
C. R2
D. R3
E. R4
F. R5
G. drop

→ E

1 pt **Q 2.4** On the path between two nodes A and B are a number of links and IPv4 routers. One of the links on this path has a very peculiar behaviour: It only supports very small packets, smaller than the MTU used in TCP in nodes A and B, but large enough to carry TCP segments with a few bytes data. Furthermore, the link drops every second packet (in both directions). Will a file larger than the MTU ever be successfully transferred from A to B?

A. In the end the file will be successfully transferred, because the transmitter will make sure it transmits segments small enough to fit on the second link, and TCP will retransmit the dropped segments.
B. In the end the file will be successfully transferred, because TCP will retransmit missing fragments until all segments have been completely received
C. In the end the file will be successfully transferred, because the IP layer in the node just before the peculiar link will continue to retransmit fragments until the IP packets can be reassembled at the IP layer in the node just after the peculiar link.
D. In the end the file will be successfully transferred, because the IP layer in node A will continue to retransmit fragments until the IP packets can be reassembled at the IP layer in node B.
E. The file will never be successfully transferred because TCP retransmits complete segments, not fragments, and the first segment will never reach B intact.

→ E

Every attempt by TCP to send an MTU-sized segment will result in (at least) two fragments to be sent over this strange link. Since every second packet, and thus every second fragment, will be dropped, the IP layer in host B will never get all fragments from a single packet, and will thus never be able to reassemble the packet for delivery to TCP. Hence answer E.
Note that retransmissions are done by TCP, while the fragmentation and reassembly are done at a lower layer, namely IP. So TCP cannot just retransmit the lost fragment (answer B), nor will IP do this (answer C or D).
Some modern TCP implementations will actually try to do path MTU discovery, and thus indeed adopt their MTU to the smallest MTU possible on the entire path (answer A). But it was not given that this particular TCP also does that; to the contrary, the text says that this TCP uses an MTU larger than the small packets of the peculiar link.

1 pt **Q 2.5** In the description of DHCP, also a DHCP relay is specified. Whereas a host is broadcasting its DHCPDISCOVER message to the relay, the relay is unicasting it to the server. Why is that?

A. to avoid looping of DHCP messages
B. because the DHCP relay knows the IP address of the DHCP server
C. because the message from the DHCP relay to the DHCP server is sent using TCP instead of UDP
D. because the DHCP relay has learnt the IP address of the server from the ARP protocol
E. to avoid that the forwarded DHCPDISCOVER message will be received by multiple DHCP servers

→ B

1 pt **Q 2.6** One of the fields in a DNS Resource Record is the TTL field. What is the purpose of this field?

A. to force caching servers and clients to query the authoritative name server for a domain periodically while the resource record is still needed
B. to inform a host to which value the TTL field in an IP packet should be set to ensure the corresponding domain is reached
C. to avoid that DNS queries are being forwarded in the network for a long time, e.g. due to routing loops
D. for load distribution, to inform the DNS client for how long the resource record can be used before moving on to the next record in the list
E. to inform the DNS client from which moment on the Resource Record for a new domain will be valid
F. to inform the owner of the domain when the domain name registration is due to be renewed.

→ A

Many chose answer C, but that is wrong.
Of course, the DNS request or reply will be transported inside an IP packet, and IP will have a TTL field in its header which has the purpose described by answer C.
But the question was not about the TTL field in the IP header, but about the TTL field in the DNS Resource Record.
The purpose of that field is to indicate how long the information is to be considered valid, before re-requesting it.

2 pt **Q 2.7** Consider a router that receives an IP multicast packet on one of its incoming links. Using PIM-SM the router has somehow learned that this packet needs to be duplicated and forwarded to routers on two outgoing links. Mark all fields in the headers of the two **outgoing copies** of the same packet that are **different**.

A. source MAC address
B. destination MAC address
C. TTL
D. source IP address
E. destination IP address
F. none, all fields are the same

→ A,B

The MAC addresses are different for each outgoing link, as they have only local significance on each LAN. The router itself also has a different MAC address on each interface, so also the source MAC address will be different.
TTL will be decremented at every hop, but will be the same in the two outgoing packets.
Source IP will be the same; it indicates the original sender of the packet, so no reason to change it.
Destination IP will be the same; in IP multicast, the destination IP indicates to which group *of hosts the packets should be sent, so it is the same in all outgoing copies.*

---

3. Routing

Link state routing

Please consider the following network, in which 5 routers, with names A, B, C, D, and E are connected with bi-directional links with the following costs:

A - C = 3;

A - D = 1;

A - E = 30;

B - D = 5;

B - E = 1;

C - E = 1;

D - E = 4.

Assume that the network is running a link-state routing protocol, using Dijkstra's shortest path algorithm for route calculation. Calculate the optimal routes from the point of view of node A using the Dijkstra algorithm. We recommend you do the algorithm on paper first.

Below, we ask you to fill in the "confirmed" and the "tentative" list after every step. Use the same format as in the book, so a sequence of (X,n,Y), where X is the destination, n the cost of the currently cheapest known path the X, and Y the next hop on that path.

We start with the confirmed list in the first step, which is clearly (A,0,-) .

4 pt **Q 3.1** In the next step, the first non-empty tentative list is calculated. Please give this list.

→ (C,3,C)(D,1,D)(E,30,E)

Q 3.2 Subsequently, the confirmed list is updated. Please give this list (according to the numbering

of steps in the book, this is step 3).

→ (A,0,-)(D,1,D)

Q 3.3 Give the next tentative list, after recalculating it (step 4 according to the book's numbering).

→ (B,6,D)(C,3,C)(E,5,D)

Q 3.4 Give the updated confirmed list (step 5).

→ (A,0,-)(D,1,D)(C,3,C)

Q 3.5 Give the recalculated tentative list (step 6).

→ (B,6,D)(E,4,C)

Q 3.6 Give the next confirmed list (step 7).

→ (A,0,-)(D,1,D)(C,3,C)(E,4,C)

Q 3.7 Give the next tentative list (step 8).

→ (B,5,C)

Q 3.8 Give the final confirmed list, from which the entries in the forwarding table can be derived.

→ (A,0,-)(D,1,D)(C,3,C)(E,4,C)(B,5,C)

Distance Vector routing

Consider the same network as above, now assuming the use of a distance vector routing algorithm, with all nodes doing their updates simultaneously. For now, no split horizon or poisoned reverse is used. Let us now assume that in the initial iteration all nodes will send a packet to their neighbours in which they indicate that they can reach themselves with cost 0, so, e.g., node A will send (A, 0) to C, D, and E.

3 pt **Q 3.9** Which is the list of costs that A will send to C (and its other neighbours) in the next iteration, after receiving the messages from the initial iteration from its neighbours? Please use as notation for this list of costs a list of (X,c), where X is the name of the node and c is the cost to X just like it is done in the rest of this question.

→ (A, 0) (C,3) (D,1) (E,30)

In the same iteration, A receives new lists of costs from its neighbours.

Q 3.10 Which is the list of costs that A will send to C in the iteration thereafter?

→ (A,0)(B,6)(C,3)(D,1)(E,4)

Q 3.11 Which list of costs will A send to C when the distance vector algorithm has converged?

→ (A,0)(B,5)(C,3)(D,1)(E,4)

Q 3.12 Same question, but now assuming split horizon is used.

→ (A,0)(D,1)

Next, assume that the link between C and E breaks.

1 pt **Q 3.13** Assuming no split horizon / poison reverse is used, what will be the cost to E reported (advertised) by node C in the next iteration?

→ 7

1 pt **Q 3.14** How many iterations will it take until A knows the new lowest cost path to E, including the one from the previous question?

→ 1

Q 3.15 Explain your answer

→ In that same iteration that Q3.13 was about, node B will report to node A that it can reach node E at a cost of 5. So node A will immediately choose the path via node B, which is now the new lowest cost path.

Note that although the question looks like the count-to-infinity questions from previous exams, in this case such counting doesn't really happen, as the new best path is known immediately.

Ad Hoc Routing

1 pt **Q 3.16** In AODV the cost to reach a certain destination is advertised in RREP messages, in a way similar to the messages with costs known destinations in regular distance-vector routing. Why are also RREQ messages used in AODV, in addition to these RREP messages?

A. to trigger the transmission of RREP messages, as these are only transmitted if data needs to be transmitted to a destination;
B. to maintain routes to all nodes in the network, even if they are moving relative to other nodes;
C. to advertise paths to the node transmitting the RREQ to all nodes in the network;
D. to periodically trigger updated RREP messages to be sent;
E. to request an on-demand route if RREP messages got lost in the network;
F. to acknowledge the correct receipt of a RREP message;

→ A

1 pt **Q 3.17** Assume an ad-hoc network using the AODV routing protocol, in which all N nodes (routers) are within each other's communication range. How many RREQ and RREP packets will be sent (in total, by all nodes in the network) for the transmission of a data packet, if the routing tables of the nodes are initially empty.

A. 0 RREQs and 0 RREPs
B. 1 RREQ and 1 RREP
C. $N-1$ RREQs and $N-1$ RREPs
D. N RREQs and N RREPs
E. 1 RREQ and N RREPs
F. 1 RREQ and $N-1$ RREPs
G. $N-1$ RREQs and 1 RREP
H. N RREQs and 1 RREP

→ G

One node transmits the initial RREQ. All other nodes hear this RREQ, and all except one will retransmit it, in an attempt to make it reach its destination. The exception is the destination node itself: it will not retransmit the RREQ, but instead it will send an RREP reply. The RREP contains the entire path back to the original source: in this case that's a direct, one-hop path, so the RREP does not need to be retransmitted by the other nodes.

---

4. IPv6

Check the following IPv6 addresses (or lookalikes) for correctness and if correct, for being in subnets.

0.6 pt **Q 4.1** 1100::3223

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 1111::0/8 and also not in 1111::0/64
C. valid IPv6 address, in the subnet 1111::0/8 but not in 1111::0/64
D. valid IPv6 address, in the subnet 1111::0/64 but not in 1111::0/8
E. valid IPv6 address, both in the subnet 1111::0/8 and in 1111::0/64

→ C

0.6 pt **Q 4.2** 1111::3223

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 1111::0/8 and also not in 1111::0/64
C. valid IPv6 address, in the subnet 1111::0/8 but not in 1111::0/64
D. valid IPv6 address, in the subnet 1111::0/64 but not in 1111::0/8
E. valid IPv6 address, both in the subnet 1111::0/8 and in 1111::0/64

→ E

0.6 pt **Q 4.3** 1111:2345::3223

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 1111::0/8 and also not in 1111::0/64
C. valid IPv6 address, in the subnet 1111::0/8 but not in 1111::0/64
D. valid IPv6 address, in the subnet 1111::0/64 but not in 1111::0/8
E. valid IPv6 address, both in the subnet 1111::0/8 and in 1111::0/64

→ C

0.6 pt **Q 4.4** 1111::2345::3223

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 1111::0/8 and also not in 1111::0/64
C. valid IPv6 address, in the subnet 1111::0/8 but not in 1111::0/64
D. valid IPv6 address, in the subnet 1111::0/64 but not in 1111::0/8
E. valid IPv6 address, both in the subnet 1111::0/8 and in 1111::0/64

→ A

0.6 pt **Q 4.5** ::1111:3223

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 1111::0/8 and also not in 1111::0/64
C. valid IPv6 address, in the subnet 1111::0/8 but not in 1111::0/64
D. valid IPv6 address, in the subnet 1111::0/64 but not in 1111::0/8
E. valid IPv6 address, both in the subnet 1111::0/8 and in 1111::0/64

→ B

2 pt **Q 4.6** Consider a network using 4 bit addresses, in which 8 hosts are allocated. Calculate the HD ratio. Outcome:

→ 0.75

Q 4.7 Give your calculation:

→ $\log_2(8) / \log_2(2^4)$

1 pt **Q 4.8** What does it mean when the HD ratio in a network is 2 ?

A. there are more hosts than network addresses.
B. there are more network addresses than hosts.
C. half of the address bits are in use.
D. half of the addresses are in use.
E. structured assignment is needed to address all hosts.

→ A

1 pt **Q 4.9** What can tunneling be used for?

A. Letting an IPv4-only client talk to an IPv6-only server.
B. Letting an IPv6-only client talk to an IPv4-only server.
C. Connecting two IPv6-only hosts via an IPv4-only network.
D. Addressing more than 2^{32} computers with IPv4.

→ C

2 pt **Q 4.10** Suppose an IPv6 host receives several fragments of packets. What information will it use to find out which fragments belong to the same packet? Mark all that apply.

A. This is impossible as IPv6 has no fragmentation.
B. Order of arrival of the fragments.
C. Source and destination address in the IPv6 header.
D. Identification field in the IPv6 header.
E. Identification field in the fragmentation header.

F. Identification field in the routing header.
G. Source and destination port number in the TCP header.
H. None of the above

→ C,E

Many chose also D, but there is no field called 'identification' in the IPv6 header (in contrast to IPv4).
Also, many chose G. However, not every packet contains a TCP header (e.g., UDP), so the reassembly process cannot rely on fields from the TCP header.

1 pt **Q 4.11** What is the point of Software Defined Networks?

A. Nothing new, software has defined networks forever.
B. Separating control software from the switches themselves.
C. Implementing parts of switches in software.
D. Using software to design a network.
E. Using software to find shortest paths in networks.
F. Using software to process packets in routers and switches.

→ B

---

5. Transport protocols

1 pt **Q 5.1**

What role does a "well-known" port number play when a client connects to e.g. a web or mail server?

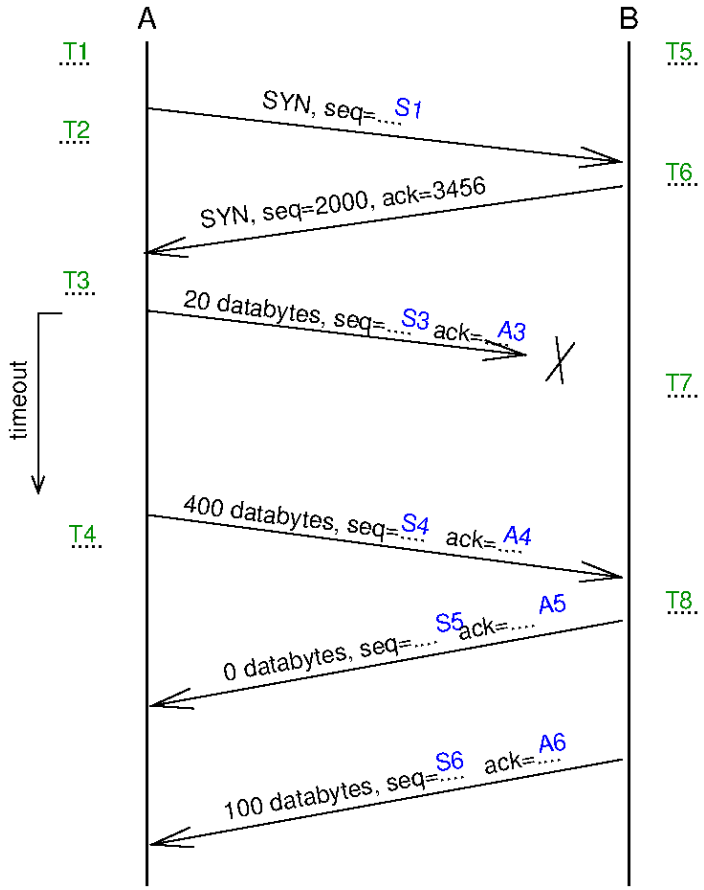
A. It is used as the source port number of the initial SYN packet.
B. It is used as the destination port number of the initial SYN packet.
C. It is used as both the source and destination port number of the initial SYN packet.
D. It is used for asking DNS what the correct port number of the server is.
E. It is obtained from DNS, which always runs on port 53.
F. None; well-known ports only play a role in UDP.

→ B

1 pt **Q 5.2** How does a server choose the port numbers in its response to a client's SYN packet?

A. Same as in the received SYN.
B. Same as in the received SYN, but interchanged.
C. Source is random, destination is same as source in SYN.
D. Source is random, destination is same as destination in SYN.
E. Destination is random, source is same as source in SYN.
F. Destination is random, source is same as destination in SYN.
G. Both are random.

→ B



The above shows a time-sequence diagram. Please fill in what should be written on the dots. T1...T8 are state names from the TCP state transition diagram. S1...S6 and A3...A6 are sequence and acknowledgement numbers.

2 pt **Q 5.3** T1 → CLOSED

Q 5.4 T2 → SYN_SENT

Q 5.5 T3 → ESTABLISHED

Q 5.6 T4 → ESTABLISHED

Q 5.7 T5 → LISTEN

Q 5.8 T6 → SYN_RCVD

Q 5.9 T7 → SYN_RCVD

Q 5.10 T8 → ESTABLISHED

1 pt **Q 5.11** S1 → 3455

Q 5.12 S3 → 3456

Q 5.13 A3 → 2001

2 pt **Q 5.14** S4 → 3456

Q 5.15 A4 → 2001

Q 5.16 S5 → 2001

Q 5.17 A5 → 3856

Q 5.18 S6 → 2001

Q 5.19 A6 → 3856

Some points of attention:

- *At T7, nothing happens. It's the time at which the third packet would have arrived at B if it had not gone lost. But it has gone lost, and B is not clairvoyant, so it doesn't know about this, and thus it has no reason to change its state at this moment.*
- *The fourth packet must be a retransmission of the third, since it is apparently triggered by a timeout. Thus, it contains the same data as the third packet, and apparently some more (see the next question). Its sequence number S4 must thus also be the same as S3.*
- *A5 = S4+400, since the fifth packet acknowledges all 400 bytes contained in the fourth packet.*
- *A6 = A5, since no new data has arrived in the meantime, so there's nothing new to be acknowledged.*
- *S6 = S5, since the fifth packet contained no data, so it doesn't occupy any place in the "sequence number space".*

1 pt **Q 5.20** In the above, you see that the retransmission of the third packet contains more bytes than the original (400 instead of 20). This must be due to the application layer handing 380 extra bytes to TCP somewhere during the timeout. Why didn't TCP transmit those extra 380 bytes in a separate packet immediately, while the timeout was still running?

A. This was not allowed by the congestion window.
B. This was not allowed by the receive window.
C. This is due to Nagle's algorithm.
D. This is to avoid the Silly Window Syndrome.
E. This is a bug.
F. Transmitting a new packet while the timeout is running, is not allowed.

→ C

A is wrong because between the third and the fourth packet, nothing happens which could increase the congestion window; also, the initial congestion window of TCP is at least one maximum-size packet.
Same for B: the receive window is communicated from host B to host A in a packet (it's a field in the TCP header); since no packeta from host B arrive at host A between the 3rd and the 4th packet, A's knowledge of B's receive window cannot have changed. So if transmitting the extra 380 bytes is allowed by the receive window at the time of the fourth packet, it was also allowed already at an earlier time.
C is correct. The Nagle algorithm says (roughly – see the book for details) that one shouldn't send multiple small packets in one RTT.
D is wrong. Silly window syndrome is a different problem, related to receive window updates, but there are no receive window updates happening here; see the book for details.
It's not a bug (option E) because the Nagle algorithm gives a good explanation for this behaviour (see C).
F is simply an incorrect statement. Remember that we needed a sufficiently large SWS (Send Window Size) in reliable data transfer, in order to be able to fully utilize a link? If F were right, TCP's send window would effectively be just 1 packet, giving very bad performance.

1 pt **Q 5.21** Why does DNS run over UDP rather than TCP?

A. DNS is a very old protocol, TCP didn't exist yet back then
B. TCP requires DNS, so if DNS runs on TCP, we'd have a problem
C. to make DNS faster by saving the connection setup overhead
D. DNS is a non-elastic application
E. this is wrong, DNS runs directly on top of IP without UDP

→ C

1 pt **Q 5.22** Sequence number wrap-around in TCP is solved by sending timestamps. How's that possible?

A. Packets are sent at fixed times, so timestamps effectively are sequence numbers.
B. Timestamps allow distinguishing between old and new packets.
C. The next packet with the same sequence number can be sent at appropriate times.

D. Timestamps allow opening the window for a longer time.
E. The wrap-around is postponed until a time when it causes no harm.

→ B

1 pt **Q 5.23** Suppose the window-scaling option is used to scale the advertised window by a factor of e.g. 16. Which of the following is true?

A. As a consequence, data must always be sent in multiples of 16 bytes.
B. As a consequence, data must always be sent in multiples of 16 packets.
C. If the receiver's real available buffer space is not a multiple of 16 bytes, it should round this value down for advertising.
D. If the receiver's real available buffer space is not a multiple of 16 bytes, it should round this value up for advertising.
E. If the receiver's real available buffer space is not a multiple of 16 packets, it should round this value down for advertising.
F. If the receiver's real available buffer space is not a multiple of 16 packets, it should round this value up for advertising.

→ C

First of all, TCP's receive window is expressed in bytes, not packets, so answers B, E and F don't make sense.
The goal of the receive window is to prevent the sender from sending more data than the receiver has space for.
The sender is not obliged to precisely fill the receive window, so answer A is also not right.
Due to the window-scaling, the receiver can only advertise window sizes that are a multiple of 16 bytes (in this example). In order to still prevent overrunning its buffer, the buffer size should round down*; if it would round up, there's a risk the sender would send to much.*
E.g., if the receiver has space for 163 bytes, it should advertise 10 × 16 = 160 bytes (round down), not 11 × 16 = 176 bytes (round up), as the latter would allow the sender to send e.g. 171 bytes, which would overrun the buffer.

Grade calculation

The grade was calculated using the following formula:

$$\text{grade} = \frac{\text{points} - 3.96}{46 - 3.96} \times 9 + 1$$

46 is the maximum number of points for this test.
3.96 is the "guessing factor": it's the number of points one would get on average from giving totally random answers at the multiple-choice questions.