Network Systems (201600146/201600197), Test 2

April 9, 2020, 13:45–16:45

(Taken together with test 3, online at home because of the corona pandemic)

The correct answers are marked with an arrow →; text on grey background gives further explanation.

1. Switching and IPv4

1 pt **Q 1.1** Consider a virtual circuit network. What does the number of distinct VCIs required in such a network depend on?

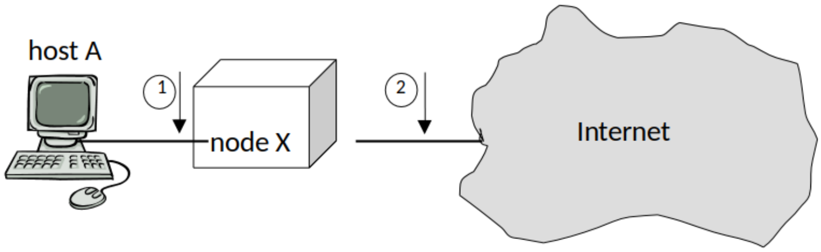
A. the number of nodes in the network
B. the number of links in the network
C. the total number of VCs in the network
→D. the number of VCs on the busiest link in the network
E. the length of the longest path in the network

The VCI identifies one circuit on a link. Since they are only unique per link, and can be reused at other links, we only need to have enough of them to fit the number of circuits on the busiest link in the network.

1 pt **Q 1.2** It is said that in a virtual circuit network, a routing algorithm is still needed. Why is that?

A. to fill the VC tables of the switches
→B. to be able to send the VC connection request to the correct next-hop switch
C. to determine the correct path from incoming interface to outgoing interface inside a switch
D. for packets for which no outgoing VC is known
E. to route between the different VCs on a link

Please consider the following network configuration. Host A is connected to the Internet, through some node X of which we do not know the functionality. We would like to deduce the functionality of node X by inspecting packets that traverse node X from A to the Internet, i.e., by looking at the same packet (with possibly somewhat modified headers) on point 1 and point 2.



1 pt **Q 1.3** Suppose the source MAC address of the packet we inspect is different at point 1 and point 2. What can we deduce?

A. node X is a switch.
B. node X is a router.
C. node X is a NAT
D. node X is a switch or a router
E. node X is a switch or a NAT
→F. node X is a router or a NAT
G. nothing (none of the above)

The MAC address does not change when a packet is forwarded by a switch or bridge, but it is changed by routers and NATs.

1 pt **Q 1.4** Suppose the source IP address of the packet we inspect is different at point 1 and point 2. What can we deduce?

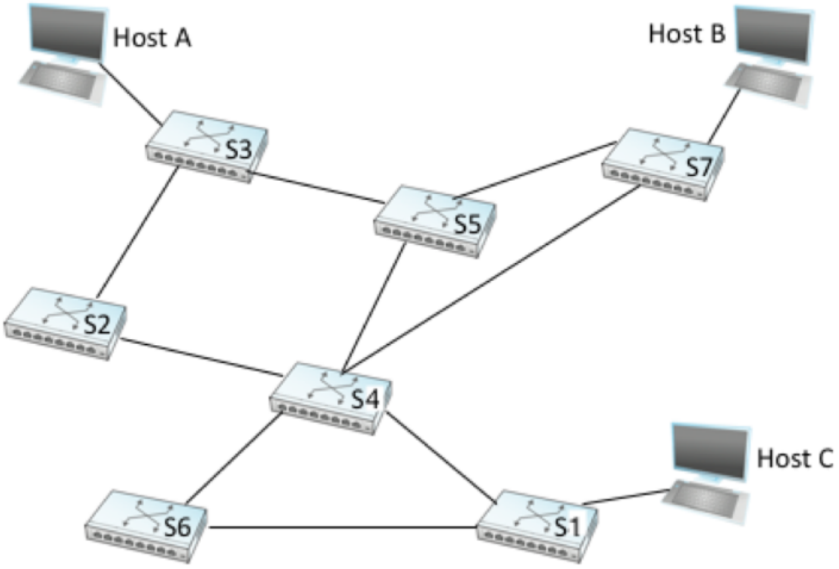
A. node X is a switch.
B. node X is a router.
→C. node X is a NAT
D. node X is a switch or a router
E. node X is a switch or a NAT
F. node X is a router or a NAT
G. nothing (none of the above)

1 pt **Q 1.5** Suppose the destination TCP port number of the packet we inspect is the same at point 1 and point 2. What can we deduce?

A. node X is a switch.
B. node X is a router.
C. node X is a NAT
D. node X is a switch or a router
E. node X is a switch or a NAT
F. node X is a router or a NAT
→G. nothing (none of the above)

Switches and routers will never touch the TCP header. NATs may change the TCP port number, but they may also not change it, depending on the situation. See the examples in the book. Since we can't be sure about the NAT, we can't draw a conclusion here.

Please consider the network of switches below.



2 pt **Q 1.6** After running the spanning tree algorithm in this network, which links are not used anymore for forwarding packets on?

A. S1 - S4
B. S1 - S6
C. S2 - S3
D. S2 - S4
→E. S3 - S5
F. S4 - S5
→G. S4 - S6
H. S4 - S7
→I. S5 - S7

S1 one will be the root; S4 and S6 have a direct link to S1, so S4-S6 is not needed (and would form a loop). From S3 to the root, there are two equally long paths; then the one with the lowest neighbour ID is chosen for use, so S2-S3, leaving S3-S5 unused. From S7 to the root, the shortest path is via S4, so S5-S7 is unused.

2 pt **Q 1.7** For some reason, the switch S3, depicted in the figure above has accidentally also been assigned identity S1 (so, there are now 2 switches S1). After running the spanning tree algorithm in this network, which links are not used anymore for forwarding packets on?

A. S1 - S4
B. S1 - S6
C. S2 - S3 (now also numbered S1)
→D. S2 - S4
E. S3 (now also numbered S1) - S5
→F. S4 - S5
→G. S4 - S6
H. S4 - S7
→I. S5 - S7

Now there are two switches broadcasting 1 as their identifier. The other switches don't know there are two S1's; they'll just try to get the shortest path to any switch that pretends to be S1. Thus, S4, S6, S2 and S5 are directly attached to one of them; S7 can choose two now equally long paths (via S5 to S3 or via S4 to S1) and will choose the one that has the lowest ID.

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

Prefix / Length	Next Hop
130.89.130.144/28	Interface 0
130.89.130.144/30	Interface 1
130.89.130.160/27	R2
130.89.130.0/24	R3
129.98.96.0/20	R4
0.0.0.0/0	R5

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.

1 pt **Q 1.8** 130.89.130.144

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

1 pt **Q 1.9** 130.89.130.191

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

1 pt **Q 1.10** 129.98.115.33

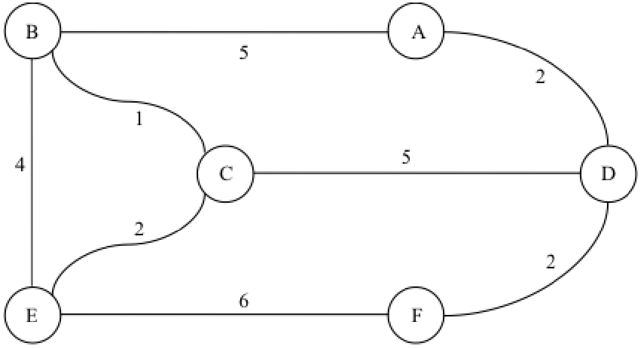
A. Interface 0
B. Interface 1
C. R2

D. R3
E. R4
→F. R5
G. drop

2. Routing

Link state routing

Please consider the following network, in which all routers (vertices) are labelled by a capital letter, and all (bi-directional) links (edges) between routers are labelled with the cost of that link. 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,-) .

Q 2.1 In the next step, the first non-empty tentative list is calculated. Please give this list.

→ (B,5,B)(D,2,D)

Q 2.2 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,2,D)

Q 2.3 Give the next tentative list, after recalculating it (step 4 according to the book's numbering).

→ (B,5,B)(C,7,D)(F,4,D)

Q 2.4 Give the updated confirmed list (step 5).

→ (A,0,-)(D,2,D)(F,4,D)

Q 2.5 Give the recalculated tentative list (step 6).

→ (B,5,B)(C,7,D)(E,10,D)

Q 2.6 Give the next confirmed list (step 7).

→ (A,0,-)(D,2,D)(F,4,D)(B,5,B)

Q 2.7 Give the next tentative list (step 8).

→ (C,6,B)(E,9,B)

Q 2.8 Give the next confirmed list (step 9).

→ (A,0,-)(D,2,D)(F,4,D)(B,5,B)(C,6,B)

Q 2.9 Give the next tentative list (step 10)

→ (E,8,B)

Q 2.10 Give the final confirmed list, from which the entries in the forwarding table can be derived.

→ (A,0,-)(D,2,D)(F,4,D)(B,5,B)(C,6,B)(E,8,B)

6 pt *In total, questions 2.1. . . 2.10 were worth 6 points.*
In the grading, we started with 6 points, and subtracted something for every error, without going negative of course.
How much was subtracted for each error depends on the seriousness of the error, and on whether you had made the same type of error before.

There was some confusion about the numbering of the steps. The description in the book counts updating the tentative list and the confirmed list each as a step; this is the numbering used above. But in some of the examples and old tests we have counted updating the tentative and confirmed list together as one step. During the test we made an announcement to clarify this.

Distance Vector routing

We consider a distance vector routing algorithm without split horizon or poison reverse. We consider a node A, which has links to two other nodes: a link to node B with link cost 2, and a link to node C with cost 1. Let us now assume that node A receives a packet from B containing the following list of costs: (B, 0), indicating that B currently only knows a path to itself at cost 0. Likewise, A will receive from C: (C, 0).

1 pt **Q 2.11** Which is the list of costs that A will send back to B and C? 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)(B,2)(C,1)

Subsequently, A receives a new list of costs from B: (B,0)(A,2)(C,100)(D,5). Furthermore, it receives from C: (C,0)(A,1)(B,100)(D,200).

1 pt **Q 2.12** Which is the list of costs that A will now send to B and C?

→ (A,0)(B,2)(C,1)(D,7)

1 pt **Q 2.13** Which list of costs will B send out to A (and its other neighbours), after it has received A's list from the previous question?

→ (B,0)(A,2)(C,3)(D,7)

Now, let us suppose that A detects that its link to C has disappeared. A sends a message to B with a list of costs reflecting this change.

1 pt **Q 2.14** How many packets with lists of cost (each resulting in an update from A to B) will B send to A **before** the real distance from B to C is given in B's list of costs?

→ 24

1 pt **Q 2.15** Please explain how you got to that answer:

→ B tells A it knows C at a cost of 3. Then A tells B it knows C at a cost of 5. Then B tells A it knows C at a cost of 7. And so on. From an earlier message we know that B has a direct link to C with a cost of 100, so the process will stop when A tells B a cost of more than 100. Counting carefully, we see B sends 24 updates to A before this happens.

2 pt *Q2.14 and 2.15 together were worth 2 points.*

We accepted a number of variations, and subtracted part of the 2 points for various minor errors (like a calculation error, or overlooking that the costs go up by 2 in every step). On the other hand, giving the right number without *any explanation was worth 0 points.*

1 pt **Q 2.16** Why does the BGP protocol include path information in its routing messages? (mark all that apply)

→A. to avoid count-to-infinity
→B. to allow administrators to control which authoritative systems their packets are going through
C. to use for source routing
D. to use in the OSPF (Open Shortest Path First) protocol
E. for routing acks to find their way back to the source

Each correctly selected option is worth 0.5 point, but each wrongly selected option costs 0.5 point, albeit without going negative.

---

3. IPv6 and addressing

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

0.5 pt **Q 3.1** ::5005

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet ::0/32 and also not in ::0/128
→C. valid IPv6 address, in the subnet ::0/32 but not in ::0/128
D. valid IPv6 address, in the subnet ::0/128 but not in ::0/32
E. valid IPv6 address, both in the subnet ::0/32 and in ::0/128

0.5 pt **Q 3.2** 0001:0001::5005

A. not a valid IPv6 address
→B. valid IPv6 address, not in the subnet ::0/32 and also not in ::0/128
C. valid IPv6 address, in the subnet ::0/32 but not in ::0/128
D. valid IPv6 address, in the subnet ::0/128 but not in ::0/32
E. valid IPv6 address, both in the subnet ::0/32 and in ::0/128

0.5 pt **Q 3.3** 0100::0200:5005

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 0100::0/32 and also not in 0100::/128
→C. valid IPv6 address, in the subnet 0100::0/32 but not in 0100::/128
D. valid IPv6 address, in the subnet 0100::0/128 but not in 0100::/32
E. valid IPv6 address, both in the subnet 0100::0/32 and in 0100::/128

0.5 pt **Q 3.4** 0100::0200::5005

→A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 0100::0/32 and also not in 0100::/128
C. valid IPv6 address, in the subnet 0100::0/32 but not in 0100::/128
D. valid IPv6 address, in the subnet 0100::0/128 but not in 0100::/32
E. valid IPv6 address, both in the subnet 0100::0/32 and in 0100::/128

0.5 pt **Q 3.5** 987f:0200::5005

A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 987f::0/18 and also not in 987f::/22
C. valid IPv6 address, in the subnet 987f::0/18 but not in 987f::/22
D. valid IPv6 address, in the subnet 987f::0/22 but not in 987f::/18
→E. valid IPv6 address, both in the subnet 987f::0/18 and in 987f::/22

0.5 pt **Q 3.6** 987f:0200:1:2:3:4:5005

→A. not a valid IPv6 address
B. valid IPv6 address, not in the subnet 987f::0/18 and also not in 987f::/22

C. valid IPv6 address, in the subnet 987f::0/18 but not in 987f::/22
D. valid IPv6 address, in the subnet 987f::0/22 but not in 987f::/18
E. valid IPv6 address, both in the subnet 987f::0/18 and in 987f::/22

Q 3.7 Consider a network using 24 bit addresses, in which 1 million hosts are allocated. Calculate the HD ratio. Outcome: → 0.83

Q 3.8 Give your calculation: → $\log(10^6) / \log(2^{24})$

Questions 3.7 and 3.8 together were worth 2 points.

1 pt **Q 3.9** What does it mean when the HD ratio in a network is 0.50 ?

A. half of all addresses are in use.
B. half of the address bits are in the subnet mask.
C. half of the subnets have at least one computer on them.
D. half of the addresses are allocated to hosts, the other half to routers.
E. half as many hosts could be accomodated if structured address assignment was used.
→F. half as many address bits would suffice, if the addresses wouldn't need structured assignment.

Consider the HD ratio calculated with the base-2 logarithm. Then the denominator is the number address bits ($\log_2$ of the number of possible addresses), and the numerator is the number o bits needed to uniquely identify each host (with n bits, you can make 2^n different addresses). Thus, if the numerator is twice the denominator, apparently the network has twice as many address bits as would be needed if all addresses could be used.

1 pt **Q 3.10** How is the compatibility between IPv4 and IPv6 ?

A. IPv4 clients can connect to IPv6 servers, and IPv6 clients can connect to IPv4 servers.
B. IPv4 clients can connect to IPv6 servers, but IPv6 clients cannot connect to IPv4 servers.
C. IPv4 clients cannot connect to IPv6 servers, but IPv6 clients can connect to IPv4 servers.
→D. IPv4 clients cannot connect to IPv6 servers, and IPv6 clients cannot connect to IPv4 servers.

The IPv4 and IPv6 header formats are different. A system which only knows about IPv4 cannot produce or understand an IPv6 packet, and vice versa. It doesn't matter whether they are servers or clients.
(In practice, interoperability is often achieved by having some clients and some servers run software which can do both IPv4 and IPv6, but that was not given here.)

Suppose an IPv6 packet containing a TCP segment gets fragmented into two fragments. What would the headers in the fragments be?

1 pt **Q 3.11** First fragment:

→A. First the IPv6 header, then the fragmentation header, then the TCP header.
B. First the IPv6 header, then the TCP header, then the fragmentation header.
C. First the IPv6 header, then the TCP header (and no fragmentation header).
D. First the IPv6 header, then the fragmentation header (and no TCP header).
E. Only the fragmentation header (and no IPv6 nor TCP header).
F. Only the IPv6 header (and no fragmentation nor TCP header).

1 pt **Q 3.12** Second fragment:

A. First the IPv6 header, then the fragmentation header, then the TCP header.
B. First the IPv6 header, then the TCP header, then the fragmentation header.
C. First the IPv6 header, then the TCP header (and no fragmentation header).
→D. First the IPv6 header, then the fragmentation header (and no TCP header).
E. Only the fragmentation header (and no IPv6 nor TCP header).
F. Only the IPv6 header (and no fragmentation nor TCP header).

IPv6 does not have fragmentation fields in the standard header, so a separate fragmentation (extension) header is needed. Since this extension header contains the information about where in the whole packet this fragment fits, *it needs to be present in all fragments. On the other hand, the TCP header is (from IP's point of view) part of the higher-layer data (the payload), so there is no need to carry a copy of it in every packet.*

---

4. Transport layer protocols

Suppose a web browser has fetched an HTML file from http://www.utwente.nl/ using HTTP/1.0 (i.e., no persistent connections); the TCP connection used for this we'll call the 'first' request. Next, it fetches a picture file from that same HTTP server.

1 pt **Q 4.1** What will the port numbers in the *second* connection request be?

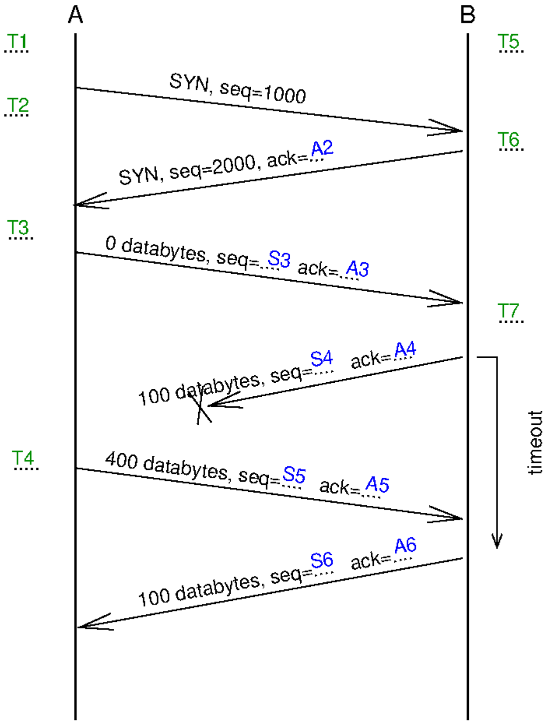
A. source and destination both 80
B. source = 80, destination is same as first request
C. destination = 80, source is same as first request
D. source = 80, destination is random and different from first request
→E. destination = 80, source is random and different from first request

Destination must again be 80, since that is the port the HTTP server is listening on. The source must be different from the previous request, otherwise packets belonging to the two connections would be confused.

1 pt **Q 4.2** In the response to that second connection request, what will the port numbers be?

A. same as in the request
→B. interchanged compared to the request
C. source = 80, destination = random
D. destination = 80, source = random

The source port of the response packet is the destination port of the request, and vice versa.


A
B
T1
T5
SYN, seq=1000
T2
T6
SYN, seq=2000, ack=A2
T3
0 databytes, seq=S3 ack=A3
T7
100 databytes, seq=S4 ack=A4
timeout
T4
400 databytes, seq=S5 ack=A5
100 databytes, seq=S6 ack=A6

The above shows a time-sequence diagram. Please fill in what should be written on the dots. T1...T7 are state names from the TCP state transition diagram. S3...S6 and A2...A6 are sequence and acknowledgement numbers. (Note: to avoid a lot of scrolling, it's probably easiest if you write them down on a piece of paper before filling them in in the input fields below.)

Q 4.3 T1 = CLOSED
Q 4.4 T2 = SYN_SENT
Q 4.5 T3 = ESTABLISHED
Q 4.6 T4 = ESTABLISHED
Q 4.7 T5 = LISTEN
Q 4.8 T6 = SYN_RCVD
Q 4.9 T7 = ESTABLISHED

2 pt *Q4.3. . . 4.9 together are worth two points.*

Q 4.10 A2 = 1001
Q 4.11 S3 = 1001
Q 4.12 A3 = 2001

1 pt *Q4.10. . . 4.12 together are worth one point.*

Note that the SYN flag itself takes 1 position in the sequence number space; that's why the ack numbers are 1 higher than the preceeding seq numbers.

Q 4.13 S4 = 2001
Q 4.14 A4 = 1001
Q 4.15 S5 = 1001
Q 4.16 A5 = 2001
Q 4.17 S6 = 2001
Q 4.18 A6 = 1401

2 pt *Q4.13. . . 4.18 together are worth two points.*

A couple of things to note that often went wrong:

- *Sequence numbers in TCP count bytes, not packets.*
- *If a packet contains no data bytes, the seq number doesn't increase for the next packet; such a packet without data bytes is effectively a pure ack packet.*
- *If a packet is retransmitted, it has the same seq number (since that indicates where in the message this segment belongs); the ack number may change if data has been received in the meantime.*

1 pt **Q 4.19** Why did Google choose to use UDP rather than TCP for the QUIC protocol?

A. reliable delivery is not important for web browsing
B. this decision was a mistake the Google engineers now regret
C. retransmissions take too much time for real-time audio and video
D. this is not correct, QUIC can also run on TCP rather than UDP, if desired
→E. so that fragments of different files that are being downloaded simultaneously don't need to wait for each other

For web browsing, you do need reliable delivery (otherwise there could be gaps or errors in the web page), and that also means retransmissions are needed, which pretty much excludes the first and third options. Nothing was said in the book about Google engineers regretting their decision, and the book does say explicitly that QUIC runs on UDP, excluding the second and fourth options. This leaves the fifth option; and indeed, section 5.2.10 of the book mentions the issue of files having to wait for each other as a problem in the use of TCP.

1 pt **Q 4.20** There are several extensions to TCP for dealing with ”long fat pipes”. Why aren't there any such extensions for UDP?

A. UDP is rarely used over long fat pipes
→B. UDP does not have sequence numbers

C. long fat pipes just cause UDP to be even more unreliable (that's what the 'U' in UDP stands for anyway)
D. the extensions designed for TCP can be used for UDP as well

Since UDP doesn't have sequence numbers to count bytes or packets, there are no issues such as sequence number wrapping (leading to possible confusion between old and new packets with the same sequence number), or a window being too small to fully use the link speed.

1 pt **Q 4.21** When we studied the link layer, we learnt that the sequence number space need not be larger than sum of the send and receive window (SWS+RWS). However, the sequence number fields in the TCP header are much larger (32 bit) than the receive window field (16 bit). Why is this?

A. The sequence numbers were made 32 bit in anticipation of the "window scaling" extension, which allows larger windows.
→B. TCP needs to deal with reordered packets, while at the link layer reordering does not happen.
C. For compatibility with IPv4's 32-bit address length.
D. TCP sequence numbers count bytes rather than packets.
E. To compensate for the relatively weak checksum that TCP uses, compared to the CRC at the link layer.

The explanation that a sequence number space of SWS+RWS is sufficient, assumes that packets cannot be reordered. If reordering is possible, all bets are off. Then you can have one packet being delayed a lot in the network and arriving at the destination much later than the packets were sent after it. If sequence numbers are reused, then clearly this can cause errors. So you need a much larger sequence number space, so large that a sequence number need not be reused for a time that is longer than the longest possible delay of a packet (typically taken to be about two minutes in the internet; the TTL field in the IPv4 header was originally counting seconds).

Many students chose option D., but this is incorrect; it's true that TCP sequence numbers count bytes, but so does the window field.

Grade calculation

Your grade was calculated using the following formula:

$$\text{grade} = \frac{\text{points} - 3.7}{43 - 3.7} \times 9 + 1$$

43 is the maximum number of points for this test.
3.7 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.