ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Computer Networks - Final Exam

December 23, 2016

Duration: 2:15 hours, closed book.

- This is a closed-book exam.
- Please write your answers on these sheets in a readable way, in English or in French.
- Please do **not** use a red pen.
- You can use extra sheets if necessary (don't forget to put your name on them).
- The total number of points is 100.
- This document contains 19 pages.
- Good luck!

Full Name (Nom et Prénom):

SCIPER No:

Division: ☐ Communication Systems ☐ Computer Science
☐ Other (mention it): . . . . . .

Year: ☐ Bachelor Year 2 ☐ Bachelor Year 3
☐ Other (mention it): . . . . . .

(answers to the questions are shown in italic and blue)

Problem 1 (10 points)

For each question, please circle a single best answer.

1. Routers operate at the:
   (a) Application layer
   (b) Transport layer
   (c) Network layer *(Correct)*
   (d) Link layer

2. Circuit switching:
   (a) Can support a higher number of users than packet switching.
   (b) Is simpler to implement than packet switching.
   (c) Offers a more efficient use of resources than packet switching.
   (d) Offers a more predictable performance than packet switching. *(Correct)*

3. A datagram with size 1250 bytes is sent over a link with rate 10 Mbps, a length of 200 km, and a signal propagation speed of 200,000 km/s. The time it takes to transfer the datagram to the other endpoint (from first bit sent to last bit received) is:
   (a) 1 ms
   (b) 1.125 ms
   (c) 2 ms *(Correct)*
   (d) 4 ms

4. If we double the transmission rate of a link:
   (a) We double the link's propagation delay.
   (b) We reduce the link's propagation delay by half.
   (c) We reduce any packet's transmission delay over that link by half. *(Correct)*
   (d) We do not affect any delay component.

5. The only type of delay experienced by a packet that can be zero is:
   (a) propagation delay
   (b) queuing delay *(Correct)*
   (c) transmission delay
   (d) processing delay

6. BitTorrent uses DHTs to:

   (a) Search for content based on user-specified keywords (e.g. ”Game of Thrones”).

   (b) Search for the addresses of peers that store a specific content based on a content identifier. *(Correct)*

   (c) Store content on many peers so that it remains available if the peers leave the network.

   (d) Cache content on many peers to reduce download times.

7. Distance-vector routing algorithms suffer from the following scaling problem as the total number of routers in the network grows:

   (a) Large storage requirements, since each router needs to store the entire network graph.

   (b) Large number of open connections, since each router needs to exchange messages with all the other routers in the network.

   (c) Long convergence time, since the number of iterations needed to reach convergence increases as the number of routers in the network increases. *(Correct)*

   (d) There are actually no such scaling problems.

8. TCP offers reliable data delivery. Why does the link layer sometimes also offer ACKs and retransmissions?

   (a) To reduce TCP’s complexity.

   (b) To increase the probability that data is successfully delivered.

   (c) To reduce the time it takes for data to be successfully delivered. *(Correct)*

   (d) For historical reasons – there is really no point in offering ACKs and retransmissions at two layers.

9. The subnet mask for the network 128.178.0.0/15 is:

   (a) 255.255.128.0

   (b) 255.255.0.0

   (c) 255.254.0.0 *(Correct)*

   (d) 128.178.0.0

10. Go-back-N offers the following advantage compared to Stop-and-wait:

    (a) Higher throughput when there is little or no packet loss. *(Correct)*

    (b) Fewer packets need to be retransmitted in case of packet loss.

    (c) The receiver can handle reordered packets better.

    (d) More reliable data delivery.

Problem 2 (33 points)

Setup:

Consider the network in Figure 1, consisting of:

- Hosts A, B, C and D
- File server E, DNS server F, web server G
- Router and NAT gateway R_1
- Router R_2
- Switches S_1 and S_2

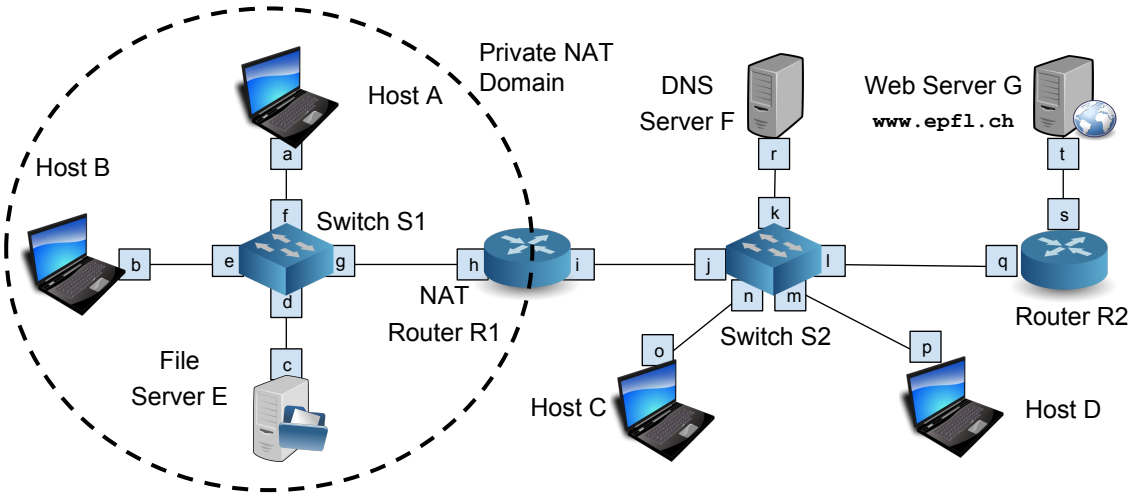

Figure 1: The Network Topology used in Section A

Interface h and all the interfaces to its left belong to a private NAT domain that uses R_1 as its NAT gateway.

Question 1 (10 points):

Allocate an IP prefix to each IP subnet and an IP address to each network interface that needs one, following these rules:

- All public addresses must be allocated from `2.2.2.0/24` (they should have the binary format `00000010.00000010.00000010.xxxxxxxx`).
- All private addresses must be allocated from `10.1.0.0/16` (they should have the binary format `00001010.00000001.xxxxxxxx.xxxxxxxx`).
- Each IP subnet must be allocated the smallest possible IP prefix.
- The first and last address in each IP subnet must be reserved for the network and broadcast address. These two addresses cannot be assigned to any interface.
- Router interfaces need IP addresses.

Use Table 1 to show the IP address allocated to each interface. Justify your answer on page 6.

Network interface	**IP address**
Example: x	1.2.3.4
a	10.1.0.1/29
b	10.1.0.2/29
c	10.1.0.3/29
h	10.1.0.4/29
i	2.2.2.1/29
o	2.2.2.2/29
p	2.2.2.3/29
q	2.2.2.4/29
r	2.2.2.5/29
s	2.2.2.9/30
t	2.2.2.10/30

Table 1: Allocation of IP addresses for the network interfaces in Figure 1

The network consists of the following subnets:

- *Subnet 1: Hosts A and B, file server E, switch S_1 and interface h of router R_1 (4 routable interfaces + 1 network address + 1 broadcast address = 6 addresses)*
- *Subnet 2: Hosts C and D, DNS server F, switch S_2, interface i of router R_1 and interface q of router R_2 (5 routable interfaces + 1 network address + 1 broadcast address = 7 addresses)*
- *Subnet 3: Web server G and interface s of router R_2 (2 routable interfaces + 1 network address + 1 broadcast address = 4 addresses)*

The other interfaces belong to switches, which do not need IP addresses, as they operate on the link layer.

A possible allocation of IP address spaces for each subnet is:

- *Subnet 1: 10.1.0.0/29 or 10.1.0.0 – 10.1.0.7*
- *Subnet 2: 2.2.2.0/29 or 2.2.2.0 – 2.2.2.7*
- *Subnet 3: 2.2.2.8/30 or 2.2.2.8 – 2.2.2.11*

We can use the binary representation to check that the allocation is correct:

`00001010.00000001.00000000.00000000` *or 10.1.0.0/29 (network addr. for Subnet 1)*
`00001010.00000001.00000000.00000001` *or 10.1.0.1/29 (interface a)*
`00001010.00000001.00000000.00000010` *or 10.1.0.2/29 (interface b)*
`00001010.00000001.00000000.00000011` *or 10.1.0.3/29 (interface c)*
`00001010.00000001.00000000.00000100` *or 10.1.0.4/29 (interface h)*
`00001010.00000001.00000000.00000101` *or 10.1.0.5/29 (not used)*
`00001010.00000001.00000000.00000110` *or 10.1.0.6/29 (not used)*
`00001010.00000001.00000000.00000111` *or 10.1.0.7/29 (broadcast addr. for Subnet 1)*
`00000010.00000010.00000010.00000000` *or 2.2.2.0/29 (network addr. for Subnet 2)*
`00000010.00000010.00000010.00000001` *or 2.2.2.1/29 (interface i)*
`00000010.00000010.00000010.00000010` *or 2.2.2.2/29 (interface o)*
`00000010.00000010.00000010.00000011` *or 2.2.2.3/29 (interface p)*
`00000010.00000010.00000010.00000100` *or 2.2.2.4/29 (interface q)*
`00000010.00000010.00000010.00000101` *or 2.2.2.5/29 (interface r)*
`00000010.00000010.00000010.00000110` *or 2.2.2.6/29 (not used)*
`00000010.00000010.00000010.00000111` *or 2.2.2.7/29 (broadcast addr. for Subnet 2)*
`00000010.00000010.00000010.00001000` *or 2.2.2.8/30 (network addr. for Subnet 3)*
`00000010.00000010.00000010.00001001` *or 2.2.2.9/30 (interface s)*
`00000010.00000010.00000010.00001010` *or 2.2.2.10/30 (interface t)*
`00000010.00000010.00000010.00001011` *or 2.2.2.11/30 (broadcast addr. for Subnet 3)*

Question 2 (14 points):

A web browser running on Host A retrieves web page `index.html` from `www.epfl.ch` (web server G). Describe all the messages that are **received or forwarded by $\mathbf{R_1}$** until the browser displays the web page, by filling Table 2. Make the following assumptions:

- All hosts use DNS server F as their local DNS server.
- All caches on all devices are empty.
- `index.html` fits in one packet and does not refer to any other objects.

Column "Trans." refers to the transport-layer protocol. Column "App." refers to the application-layer protocol. When you want to refer to the IP address of interface x, write "IP of x". When you want to refer to the MAC address of interface x, write "MAC of x". If a field is not applicable, indicate that with a "–".

#	Src MAC	Dst MAC	Src IP	Dst IP	Trans.	Src Port	Dst Port	App.	Purpose
ex	MAC of x	MAC of y	IP of w	IP of v		5000	6000	HTTP	Request: `image.png`
1	MAC of a	broadcast	–	–	–	–	–	–	ARP request: What is the MAC for the IP of h?
2	MAC of h	MAC of a	–	–	–	–	–	–	ARP reply: MAC of h
3	MAC of a	MAC of h	IP of a	IP of r	UDP	8000	53	DNS	Req. IP of `www.epfl.ch`
4	MAC of i	broadcast	–	–	–	–	–	–	ARP request: What is the MAC for the IP of r?
5	MAC of r	MAC of i	–	–	–	–	–	–	ARP reply: MAC of r
6	MAC of i	MAC of r	IP of i	IP of r	UDP	10000	53	DNS	Req. IP of `www.epfl.ch`
7	MAC of r	MAC of i	IP of r	IP of i	UDP	53	10000	DNS	Reply: IP of t
8	MAC of h	MAC of a	IP of r	IP of a	UDP	53	8000	DNS	Reply: IP of t
9	MAC of a	MAC of h	IP of a	IP of t	TCP	9000	80	HTTP	TCP SYN
10	MAC of i	broadcast	–	–	–	–	–	–	ARP request: What is the MAC for the IP of q?
11	MAC of q	MAC of i	–	–	–	–	–	–	ARP reply: MAC of q
12	MAC of i	MAC of q	IP of i	IP of t	TCP	20000	80	–	TCP SYN
13	MAC of q	MAC of i	IP of t	IP of i	TCP	80	20000	–	TCP SYN-ACK
14	MAC of h	MAC of a	IP of t	IP of a	TCP	80	9000	–	TCP SYN-ACK
15	MAC of a	MAC of h	IP of a	IP of t	TCP	9000	80	HTTP	Request: `index.html`
16	MAC of i	MAC of q	IP of i	IP of t	TCP	20000	80	HTTP	Request: `index.html`
17	MAC of q	MAC of i	IP of t	IP of i	TCP	80	20000	HTTP	Reply: `index.html`
18	MAC of h	MAC of a	IP of t	IP of a	TCP	80	9000	HTTP	Reply: `index.html`

Table 2: Messages received or forwarded by R_1, in Question 2

Note that the ARP protocol is a Link-Layer protocol, therefore is it an error to label it as a Transport-layer or Application-layer protocol.

Question 3 (3 points):

Show the NAT translation table of R_1 right after Host A has retrieved `index.html`, by filling Table 3. Use the first row of the table to specify the meaning of each column.

Local IP	Local Port	Pseudo IP	Pseudo Port
IP of a	8000	IP of i	10000
IP of a	9000	IP of i	20000

Table 3: NAT translation table of router R_1

Question 4 (3 points):

Show the forwarding tables of switches S_1 and S_2 right after Host A has retrieved `index.html`, by filling Table 4. Use the first empty row of the table to specify the meaning of each column.

Switch S_1		**Switch S_2**	
Destination MAC	Output Interface	Destination MAC	Output Interface
MAC of a	f	MAC of i	j
MAC of h	g	MAC of r	k
		MAC of q	l

Table 4: Forwarding tables of switches S_1 and S_2, in Question 3

Question 5 (3 points):

How can Host B prevent Host A from getting `index.html` by sending a single packet? When should Host B send that packet and what should it contain?

Right after host A sends a broadcast ARP-request to learn the MAC address of interface h, host B may send an ARP-reply message to A with its own MAC address. Therefore, host A will forward to B the DNS request, which will be subsequently discarded.

Problem 3 (24 points)

Question 1 (12 points):

In each of the following scenarios, Host A wants to communicate with Host B and achieve some security property. In each scenario:

i. Identify an existing problem or weakness and, if applicable, describe an attack that exploits it.

ii. Provide a solution that fixes the weakness (i.e. what should A send instead). Make sure to provide enough detail for your solution to be understandable. For example, if you say "A should use a MAC for authentication", but you do not explain how the MAC should be computed, then your answer is not complete.

Scenarios:

a. A wants to send one message m to B, ensuring confidentiality and authenticity. For this, A sends: $H(K, m)$.

b. A wants to send one message m to B, ensuring confidentiality and authenticity. For this, A sends: $K\{m\}$.

c. A wants to send one message m to B, ensuring confidentiality and authenticity. A knows B's true public key K_B^+. A sends: $K_B^+\left\{m,\ K_B^+\{H(m)\}\right\}$.

d. A is sending messages to B. Whenever B receives a message m, it should be able to verify that A indeed sent a message with the same content as m at least once. For this, A appends $H(K)$ to each message it sends.

e. A wants to send one message m to B, ensuring authenticity and data integrity. For this, A cuts m into two pieces, m_1 and m_2, sends first $[m_1,\ H(K)]$, then $[m_2,\ H(K)]$.

f. A and B are friends that want to have a sensitive online conversation, ensuring confidentiality, authenticity, and data integrity (A receives exactly the sequence of messages sent by B and vice versa). For this, they use SSL as described in class: B sends a nonce and a certificate with its public key to A, A sends a nonce and a master key to B (encrypted with B's public key), and from the master key and the nonces, they both derive other keys that they use for confidentiality and authenticity. Hint: In the example we saw in class, B was an online store. In this question, A and B are friends having a sensitive conversation.

where:

- H is a cryptographic hash function that is globally known to everyone.
- K is a symmetric key, shared between A and B.
- K_B^+ is B's public key.

(*Extra answer page for Question 1.*)

a. *i. None of confidentiality or authenticity are guaranteed, as there is no proper communication: message $H\,(K, m)$ cannot be read by B, because even if B knows the shared key K, the hash function H is an one-way function that has no inverse.*

ii. Using the available tools of the exercise setup, a solution that guarantees both confidentiality and authenticity is: A sends $K\{m,\ H\,(K, m)\}$, to provide both confidentiality and authenticity.

b. *i. The authenticity of the message cannot be guaranteed: A man-in-the-middle can change one or more bits of the ciphertext $K\,\{m\}$. B will then decrypt the message with the shared key K, but the decrypted message will be different from the one that was initially sent by A.*

ii. Using the available tools of the exercise setup, a solution that guarantees both confidentiality and authenticity is: A sends $K\{m,\ H\,(K, m)\}$.

c. *i. The authenticity of the message cannot be guaranteed: Since both the public key K_B^+ and the hash function H are publicly available, any one can generate the message $K_B^+\,\{m,\ K_B^+\,\{H(m)\}\}$.*

ii. Using the available tools of the exercise setup, a solution that guarantees both confidentiality and authenticity is: A sends $K_B^+\,\{m,\ K_A^-\,\{H(m)\}\}$.

d. *i. The authenticity of the messages cannot be guaranteed: A man-in-the-middle can change the content of one or more messages sent by A and B has no way to be sure that noone tampered with the messages, by only computing $H(K)$.*

ii. Using the available tools of the exercise setup, a solution that guarantees authenticity is the use of a MAC: i.e. A should append $H(m_i, K)$ to every message m_i that is sends.

e. *i. The authenticity and data integrity of the whole message cannot be guaranteed: A man-in-the-middle can either change the content of m_1 and/or m_2, or reorder the two messages.*

ii. A solution that guarantees both authenticity and data integrity is the use of MAC and sequence numbers: i.e. A sends first $[m_1,\ 1,\ H(K, m_1, 1)]$, then $[m_1,\ 2,\ H(K, m_1, 2)]$.

f. *i. The problem here arises from the fact that A never sends a certificate with its public key to B. Therefore, B cannot authenticate A (only A can authenticate B). A woman-in-the-middle can intercept B's message to A, pretend she is A, and carry out the entire conversation with B, pretending that she is A.*

ii. A solution is that A sends its public-key certificate to B along with the nonce and the master key.

Question 2 (12 points):
Hosts A and F secure their communication with the following protocol:

1. A sends a "hello" message with a nonce n_A and a certificate containing its public key (K_A^+)
2. F responds with a nonce n_F and a certificate containing its public key (K_F^+)
3. After this exchange, to communicate a message m_i, A sends: $K_F^+ \left\{m_i,\ K_A^- \left\{H\left(n_F, m_i\right)\right\}\right\}$
4. Similarly, to communicate a message m_j, F sends: $K_A^+ \left\{m_j,\ K_F^- \left\{H\left(n_A, m_j\right)\right\}\right\}$,

where K_A^-, K_F^- are A's and F's private keys and H is a globally known cryptographic hash function.

Questions:

a. Does this protocol guarantee confidentiality?
If yes, explain why. If not, describe an attack and provide a solution.

Yes, because of two reasons:

(i) All public and private keys are used correctly for the encryption of the messages: i.e. the public keys of the recipients of the messages are used for encryptions and the private keys of the senders are used for providing a digital signature (e.g. $K_F^- \left\{H\left(n_A, m_j\right)\right\}$) of the message.

(ii) The public keys of both communicating parties have been exchanged with certificates, which guarantees that the true and correct public keys of one are available to the other party.

b. Assume a man-in-the-middle, who records all the messages sent by A, and, when the communication between A and B ends, she resends them to F, trying to impersonate A. Will his attack be successful (or not) and why?

No, the attack cannot be successful because the nonces are correctly used inside the digital signatures of each message.

c. Is this protocol vulnerable to any attack(s) other than the ones described above?

If yes, briefly describe the attack(s) and provide the changes that make the protocol completely secure. If necessary, you can add new steps in the protocol, but **not** modify the existing ones.

Yes, it is vulnerable against the reordering attack and the deletion attacks, because no sequence numbers are used: A man-in-the-middle can reorder $K_F^+ \left\{m_i,\ K_A^- \left\{H\left(n_F, m_i\right)\right\}\right\}$ or $K_A^+ \left\{m_j \mid K_F^- \left\{H\left(n_A, m_j\right)\right\}\right\}$; or he can delete one of the messages.

*A solution to this problem could be the use of sequence numbers inside the digital signatures. However, this option is not available for this particular exercise, because we **cannot** modify the existing steps.*

Thus, a solution to this problem could be that both parties exchange a digital signature of all messages at the end of their communication. I.e. in the above protocol, we add two more steps (5 and 6):

- *5. A sends $K_F^+ \left\{K_A^- \left\{H\left(n_F, m_1 | m_2 | \ldots\right)\right\}\right\}$ to F*
- *6. F sends $K_A^+ \left\{K_F^- \left\{H\left(n_A, m_1 | m_2 | \ldots\right)\right\}\right\}$ to A*

Extra answer page for Question 2.

Problem 4 (33 points)

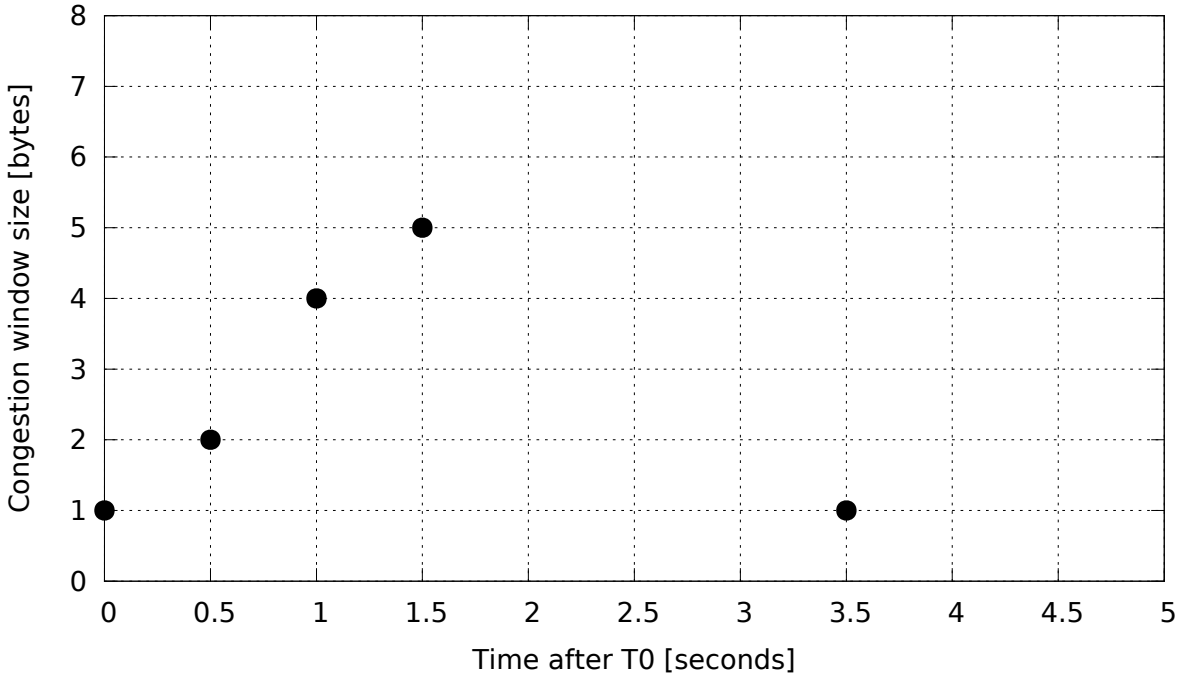

Figure 2: Congestion window of Alice over time

Alice has opened a persistent TCP connection to Bob. At time T_0, Alice starts sending to Bob, over this connection, a file of size 12 bytes in segments of MSS = 1 byte.

Figure 2 shows how the congestion window of Alice, $cwnd$, changes over time after T_0 and until the file transfer completes. Each of the $five$ points in the graph shows the time a change in $cwnd$ took place and $cwnd$'s value after the change.

Make the following assumptions:

- Transmission delay is negligible.
- Bob sends an ACK for each segment it receives.
- The first segment that Alice transmits after T_0 has sequence number 10.
- Fast-retransmit is disabled.
- Only one segment gets lost after T_0, and it is a segment sent by Alice.

Question 1 (15 points):

a. What is the RTT between Alice and Bob?

b. What is the retransmission timeout used by Alice?

c. What was the size of Alice's congestion window, $cwnd$, the last time a packet was lost before T_0?

d. Complete the diagram on the next page that shows what happens after T_0 and until the file transfer completes:

- All segments exchanged between Alice and Bob.
- The sequence numbers sent by Alice and the acknowledgment numbers sent by Bob.
- The state of Alice's congestion-control algorithm.
- The size of Alice's congestion window, `cwnd`, in bytes.
- The value of Alice's slow-start threshold, `ssthresh`, in bytes.

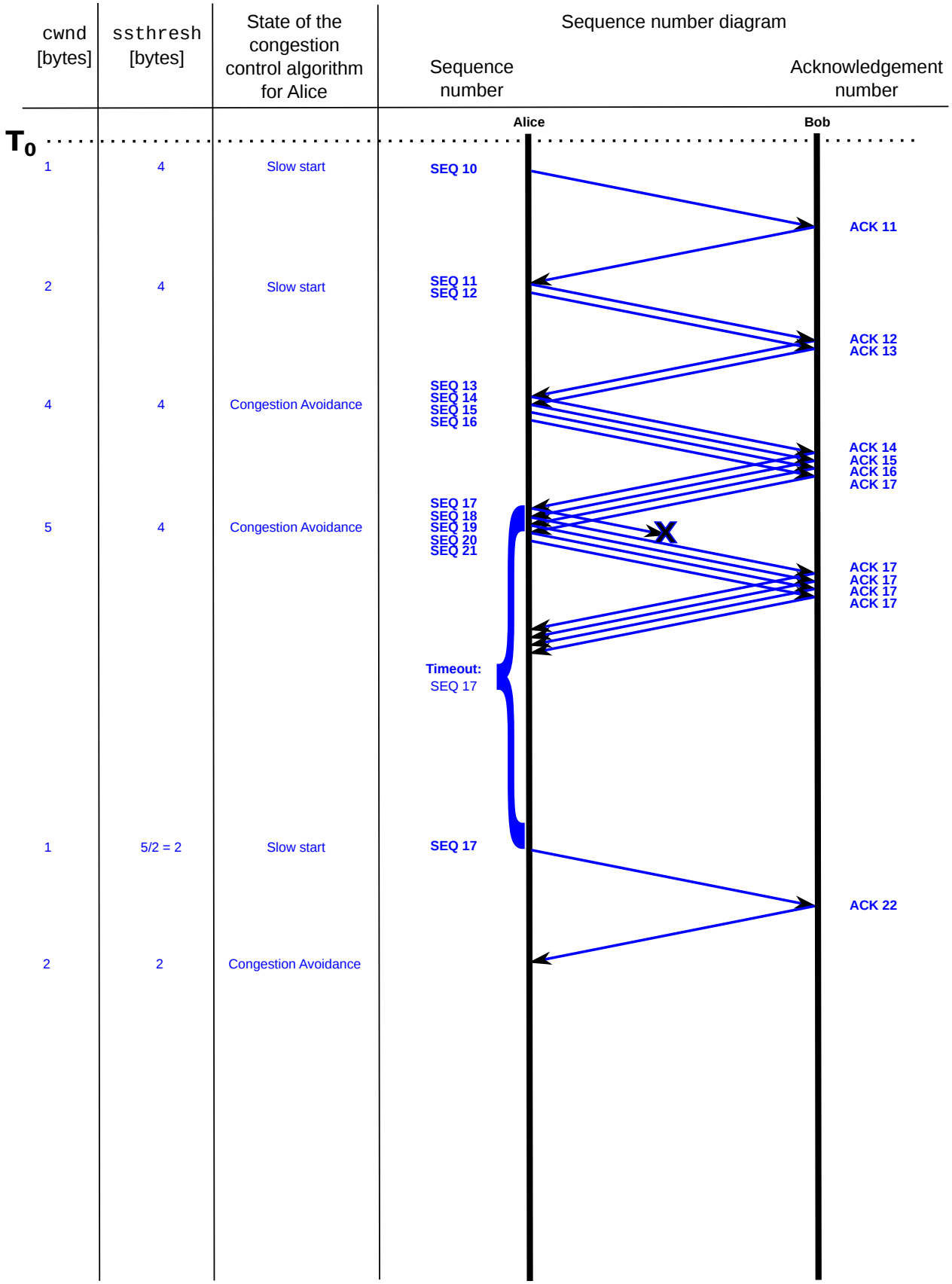

Figure 3: Sequence diagram to be completed for Question 1

(*Extra answer page for Question 1.*)

a. *The RTT between Alice and Bob is* 0.5 *seconds.*

We can extract this piece of information from what happens at time 1.5 *seconds after* T_0 *and* 2 *seconds after* T_0*. TCP is in Congestion avoidance phase during that time, and the size of the congestion window increases by* 1 *MSS (=*1 *byte) per RTT during that phase.*

Since the congestion window increases from 4 *to* 5*, between* 1.5 *secs and* 2 *secs, RTT is equal to:*

$\frac{2-1.5}{5-4} = 0.5$ *secs (2 pts)*

b. *The transmission timeout of Alice is equal to* 2 *seconds.*

According to the information we are given, there is only one packet loss (and possible cause for a time-out). This timeout, obviously, happens at time 3.5 *secs, since the window drops to size* 1 *MSS. According to the sequence diagram we have drawn, the segment that was lost was transmitted at time* 1.5*. Thus, the timeout is equal to* $3.5 - 1.5 = 2$ *secs. (2 pts)*

c. *The size of* $cwnd$ *the last time a packet was lost before* T_0 *was 8 (or 9) bytes.*

The value of ssthresh updated if and only if there is a packet loss, and it is set to $\frac{cwnd}{2}$*. Since, in our sequence diagram, the current value of ssthresh is* 4*, the last there was a packet loss (and ssthresh's value was updated),* $cwnd$ *was* $2 \cdot 4 = 8$ *bytes.*

If we assume that ssthresh has any non-integral values rounded down, another possible value could have been 9*, since* 9 *divided by* 2 *also yields* 4*. (3 pts)*

d. *See sequence diagram at Figure 3. (8 pts)*

Question 2 (6 points):

a. How long does the file transfer take? Assume that the file transfer completes once Alice has received the final ACK for file data.

b. Now assume (just for this question, i.e., Question 2b) that fast-retransmit is enabled. Does this change the duration of the file transfer and how/why?

a. From the sequence diagram, we can see that the complete file transfer takes:

- $3 \times RTT$ *(to receive the ACKs for packets* 13 *to* 16*)*
- *A timeout*
- $1 \times RTT$ *(to receive the ACK for packet* 17*, which has been retransmitted)*

Thus, in total we have $4 \cdot RTT + timeout = 4 \cdot 0.5 + 2 = 4$ *seconds. (3 pts)*

b. If Fast-Retransmit were enabled, the file transfer would take 2.5 *seconds instead.*

This is because at time 2 *seconds, Alice would have finished receiving* 3 *triple duplicate ACKs for SEQ* 17*. Thus, Alice would have retransmitted the lost packet at time* 2 *seconds, as opposed to time* 3.5 *seconds (which is the case in the sequence diagram).*

As a result, this would decrease the file transfer time by 1.5 *seconds. Thus, the total transfer time would be* $4 - 1.5 = 2.5$ *seconds. (3 pts)*

Question 3 (7 points):

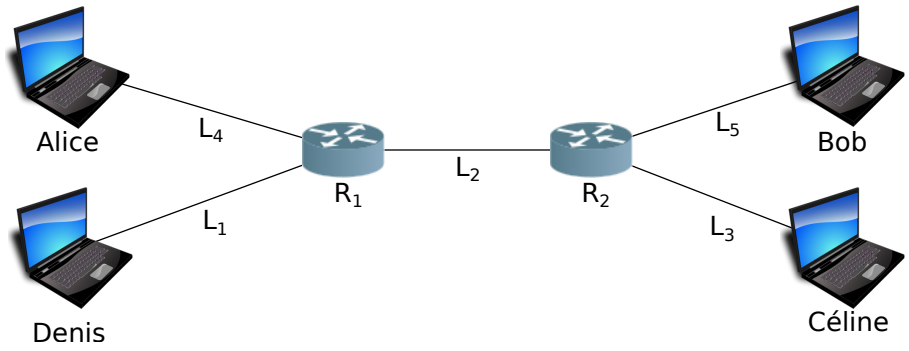

Figure 4: The Network Topology used in Question 3

In Figure 4, Alice is sending a large file to Bob using TCP. Denis tries to disrupt their communication by sending traffic to Céline. No other hosts send any traffic.

a. Describe the simplest attack strategy that achieves Denis's goal. What condition needs to hold for the transfer rates of the links such that this strategy works?

b. How will the TCP connection between Alice and Bob be affected by this attack? Draw a simple diagram that shows how Alice's congestion window, $cwnd$, evolves over time during the attack. You do not need to provide specific time values on the x-axis, just show the trend (e.g., does $cwnd$ increase monotonically?)

a. Denis prevent Alice and Bob from communicating with each other by flooding link L_2. To do so, Denis could send a constant stream of high-volume UDP traffic to Céline. (1 pt)

This attack will succeed if and only iff the capacity of L_1 is greater or equal to the capacity of L_2.

If the capacity of L_1 is greater or equal to the capacity of L_2, Denis can send traffic at a rate equal to the capacity of L_2. Since L_1 can carry Denis' attack traffic, all of it will reach L_2 at the desired rate. Thus, Denis' attack will succeed.

Otherwise, if the capacity of L_1 is lower than the capacity of L_2, the attack will fail. This is because, even if Denis sends attack traffic at a higher rate, L_1 cannot carry traffic at a higher rate than the capacity of L_1. Since the capacity of L_1 is lower than the capacity of L_2, L_2 will have some spare capacity to accommodate Alice's traffic.

Thus, we have proven that the condition we described is both necessary and sufficient. (2 pts)

b. We can see an example diagram at Figure 5 (2 pts). You can see that after the attack has commenced (sometime between 6 seconds and 8 seconds), Alice is no longer able to reliably reach Bob.

Thus, Alice's transmissions will almost always timeout.

(Not shown on the figure is how the timeout estimation algorithm of TCP will make Alice's timeout achieve increasingly bigger values, as Alices tries to determine the RTT between her and Bob. (2 pts)

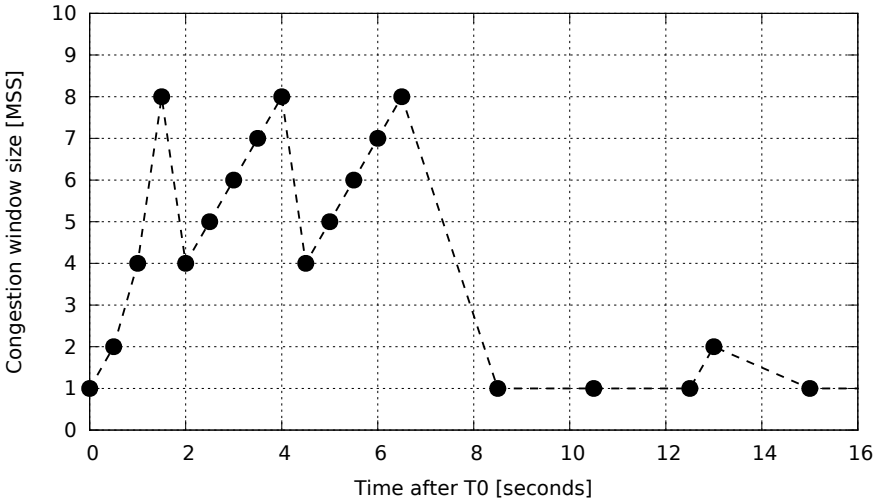

Figure 5: Congestion window of Alice over time

Question 4 (5 points):

Describe the attack strategy that achieves Denis's goal while minimizing the amount of traffic that Denis sends to Céline.

Hint: Denis does not need to send traffic at a constant rate.

The TCP algorithm makes the sender transmits its messages in bursts. As long as there is no serious congestion issues on the channel, segments will flow from the sender to the receiver in a relatively steady rate. In that state, the congestion window oscillates between $\frac{w}{2}$ and w, where w is the maximum achievable value for the congestion window..

However, when a channel becomes heavily congested (e.g., due to an ongoing attack), the sender will be experiencing constant timeouts. Thus, all traffic transmitted by the sender will be done in bursts. Moreover, these bursts will transmitted at the precise moment when the sender's timeout event triggers.

Thus, it is possible for Denis to attack the TCP flow by targetting those retransmission events. Every time that Denis expects Alice to retransmit her packets, he sends a burst of traffic to saturate the buffers of R_1. Thus, when Alice's retransmitted packet arrives at the congested router, R_1 will drop the segment of Alice, and Alice will have to wait for her timeout again. Naturally, Denis can stop transmitting flood traffic for as long as Alice is not about to transmit a new segment.

An added benefit of targetting timeout retransmissions is that such failed retransmissions cause TCP to exponentially increase its timeout interval. Thus, Alice will be sending progressively less frequently. This means that Denis will have to send even less traffic as time goes on.

What we have described so far is an attack strategy that works. In order to argue that this strategy is optimal, we need to examine whether an attack where we transmit a fewer number of messages is possible.

Note that the goal of Alice is to be able to send traffic to Bob at some predictable rate, even if this rate is significantly lower than before Denis' intervention. The goal of Denis is to make it so that Alice has no guarantee about how long it will take for Alice's segment to reach Bob.

Denis only transmits attack messages while Alice is retransmitting timed-out packets. If Denis fails to transmit enough messages to flood L_2, the segment of Alice will not get dropped, and Alice will be able to transmit a new piece of information to Bob. This will make Denis' attack ineffective. Thus, for Denis' strategy to be effective, Denis should always transmit at least as many messages are necessary to block Alice's retransmissions. Since Denis is not transmitting any attack traffic at any other point in time, it is impossible that there exists a strategy more optimal than the one we described.