ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Computer Networks - Final Exam

Prof. Argyraki

December 12, 2014

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 20 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)

1 Short questions (10 points)

For each question, please circle a single best answer.

1. A regional ISP acts as
   (a) a provider to an Internet eXchange Point (IXP).
   (b) a customer to an IXP.
   (c) a provider to a tier-1 ISP.
   (d) a customer to a tier-1 ISP. *(Correct)*

2. Which list does NOT belong with the three other lists?
   (a) transport layer, network layer, link layer.
   (b) segment, datagram, frame.
   (c) DNS, IP, Ethernet. *(Correct)*
   (d) port number, IP address, MAC address.

3. Alice uses a Go-Back-N protocol to send 10 packets to Bob. When Alice knows that Bob has received all the packets successfully, she must have received
   (a) at least 1 ACK. *(Correct)*
   (b) at most 1 ACK.
   (c) at least 10 ACKs.
   (d) at most 10 ACKs.

4. Assume host A is in TCP slow start phase (exponential increase). If host A receives an ACK for 1 MSS, the congestion window size
   (a) gets doubled.
   (b) gets halved.
   (c) gets incremented by 1 MSS. *(Correct)*
   (d) gets set to 1 MSS.

5. A system administrator manages the address space of the following networks: 128.178.4.0/23, 128.178.6.0/24, and 128.178.7.0/24. If she wants to merge the three address spaces into a single one, she can use the following network mask:
   (a) 255.255.0.0
   (b) 255.255.255.0
   (c) 255.255.240.0
   (d) 255.255.252.0 *(Correct)*

6. Internet routers use the following information in a packet's headers to make forwarding decisions:

   (a) destination MAC address.

   (b) destination IP address. *(Correct)*

   (c) destination MAC address and destination IP address.

   (d) source IP address and destination IP address.

7. Whenever a host wants to send a frame from one LAN to another we have to use a

   (a) hub.

   (b) switch.

   (c) router. *(Correct)*

   (d) any of the above is correct, they are all packet switches.

8. ARP tables provide associations of the following type:

   (a) DNS canonical name - IP address.

   (b) IP address - forwarding port number.

   (c) MAC address - forwarding port number.

   (d) IP address - MAC address. *(Correct)*

9. We can use the following protocol to provide data integrity, authentication and confidentiality:

   (a) TCP.

   (b) HTTP.

   (c) SSL. *(Correct)*

   (d) ARP.

10. Which of the following associations is NOT correct?

    (a) asymmetric key cryptography - provide confidentiality.

    (b) symmetric key cryptography - provide digital signatures. *(Correct)*

    (c) nonce - avoid replay attacks.

    (d) sequence numbers - avoid reordering attacks.

2 Problem A (30 points)

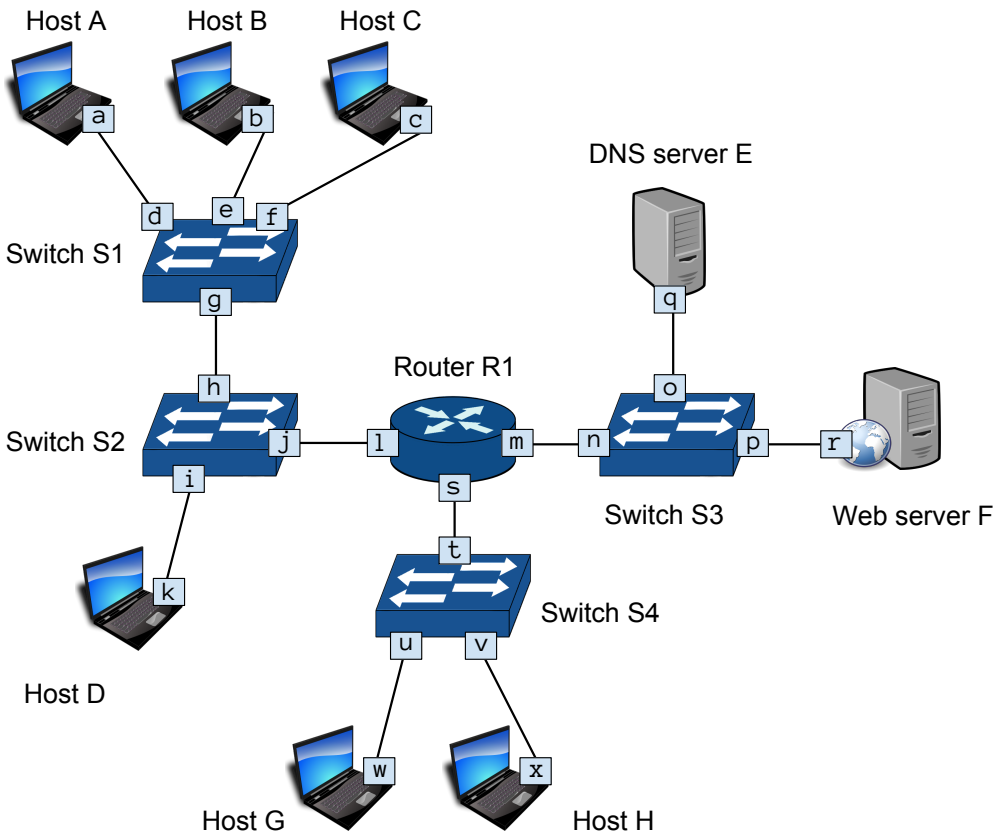

Figure 1: The network topology used in Problem A, Questions 1 and 2

Question 1 (10 points):

Consider the network in Figure 1, consisting of hosts A, B, C, D, G and H, DNS server E, web server F, router $R1$ and switches $S1$, $S2$, $S3$ and $S4$.

Assign IP addresses and network masks **only** to the network interfaces on which it is necessary, having the following constraints:

- All addresses must be allocated from `10.0.0.0/24` (they should have the binary format `00001010.00000000.00000000.xxxxxxxx`), following the basic rules for allocating IP addresses that you have learned in class.
- You should allocate the smallest possible range of IP addresses to each subnet.
- For each subnet, the network address (the first address in a subnet; for example, in `10.0.0.0/24`, the first address is `10.0.0.0`) can be assigned to an interface.
- Each host (A, B, C, D, G and H) must be able to exchange DNS messages with the DNS server E and HTTP messages with the web server F.

Answer by completing Table 1 using the format `a.b.c.d/x` if you allocate an IP address to the interface. Otherwise, write "–" if an IP address is not needed. Justify your answer on the next page.

Network interface	IP address and mask
Example: y	1.2.3.4/24
Example: z	–
a	10.0.0.0/29
b	10.0.0.1/29
c	10.0.0.2/29
d	–
e	–
f	–
g	–
h	–
i	–
j	–
k	10.0.0.3/29
l	10.0.0.4/29
m	10.0.0.8/30
n	–
o	–
p	–
q	10.0.0.9/30
r	10.0.0.10/30
s	10.0.0.12/30
t	–
u	–
v	–
w	10.0.0.13/30
x	10.0.0.14/30

Table 1: IP address allocations for the interfaces from Figure 1

Justify your answer for Question 1:

The network consists of the following subnets:

Subnet 1: Hosts A, B, C, D, switches $S1$ and $S2$, and interface l of router $R1$ (5 interfaces + 1 broadcast address = 6 addresses)
Subnet 2: DNS server E, web server F, switch $S3$ and interface m of router $R1$ (3 interfaces + 1 broadcast address = 4 addresses)
Subnet 3: Hosts G and H, switch 4 and interface s of router $R1$ (3 interfaces + 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: 10.0.0.0/29 or 10.0.0.0 – 10.0.0.7
Subnet 2: 10.0.0.8/30 or 10.0.0.8 – 10.0.0.11
Subnet 3: 10.0.0.12/30 or 10.0.0.12 – 10.0.0.15

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

`00001010.00000000.00000000.00000000` *or 10.0.0.0/29 (interface a)*
`00001010.00000000.00000000.00000001` *or 10.0.0.1/29 (interface b)*
`00001010.00000000.00000000.00000010` *or 10.0.0.2/29 (interface c)*
`00001010.00000000.00000000.00000011` *or 10.0.0.3/29 (interface k)*
`00001010.00000000.00000000.00000100` *or 10.0.0.4/29 (interface l)*
`00001010.00000000.00000000.00000101` *or 10.0.0.5/29 (not used)*
`00001010.00000000.00000000.00000110` *or 10.0.0.6/29 (not used)*
`00001010.00000000.00000000.00000111` *or 10.0.0.7/29 (broadcast addr. for Subnet 1)*
`00001010.00000000.00000000.00001000` *or 10.0.0.8/30 (interface m)*
`00001010.00000000.00000000.00001001` *or 10.0.0.9/30 (interface q)*
`00001010.00000000.00000000.00001010` *or 10.0.0.10/30 (interface r)*
`00001010.00000000.00000000.00001011` *or 10.0.0.11/30 (broadcast addr. for Subnet 2)*
`00001010.00000000.00000000.00001100` *or 10.0.0.12/30 (interface s)*
`00001010.00000000.00000000.00001101` *or 10.0.0.13/30 (interface w)*
`00001010.00000000.00000000.00001110` *or 10.0.0.14/30 (interface x)*
`00001010.00000000.00000000.00001111` *or 10.0.0.15/30 (broadcast addr. for Subnet 3)*

Question 2 (10 points):

Host A wishes to retrieve web page *index.html* from the web server F. Use Table 2 to describe all messages that are sent or received **by host *A*** until its browser displays the web page. For each message, you should specify:

- The source and destination MAC address.
- The source and destination IP address.
- The transport layer protocol (the "Proto" column).
- The source and destination port numbers.
- The message content: "*Request: ...*" or "*Reply: ...*"

Whenever you refer to the IP address of interface y, write "IP of y". Similarly, refer to the MAC address of interface y using "MAC of y". If some fields in Table 2 are not applicable, please indicate with a "–".

Assumptions:

- All devices have just been rebooted, i.e. all caches are empty.
- All hosts, servers and routers have been statically configured with an IP address, a netmask, a default gateway IP address and use DNS server E as their DNS resolver.
- Host A initially knows only the DNS name of web server F (not its IP address).
- You may ignore any TCP connection setup/closing messages.
- The web page *index.html* is very small, fitting in one packet, and does not refer to any other objects.

#	Src MAC	Dst MAC	Src IP	Dst IP	Proto	Ports	Message
-	*Example:* MAC of y	MAC of z	IP of y	IP of z	ICMP	–	Ping (echo) request
1	MAC of a	ff:ff:ff:ff:ff:ff	–	–	–	–	ARP request: What is the MAC for the IP of l?
2	MAC of l	MAC of a	–	–	–	–	ARP reply: MAC of l
3	MAC of a	MAC of l	IP of a	IP of q	UDP	1111 $\rightarrow$ 53	DNS request: What is the IP for F?
4	MAC of l	MAC of a	IP of q	IP of a	UDP	53 $\rightarrow$ 1111	DNS reply: IP of r
5	MAC of a	MAC of l	IP of a	IP of r	TCP	2222 $\rightarrow$ 80	HTTP request: GET index.html
6	MAC of l	MAC of a	IP of r	IP of a	TCP	80 $\rightarrow$ 2222	HTTP reply: 200 OK + contents of index.html

Table 2: Packets sent or received by host A in Question 2

Question 3 (10 points):

Consider the network in Figure 2. The forwarding table of all the switches are initially empty.

Host H_1 sends Ethernet frame f_1 to host H_6. Frame f_1 has source MAC address o and destination MAC address t. Host H_6 replies to host H_1 with Ethernet frame f_2. Frame f_2 has source MAC address t and destination MAC address o.

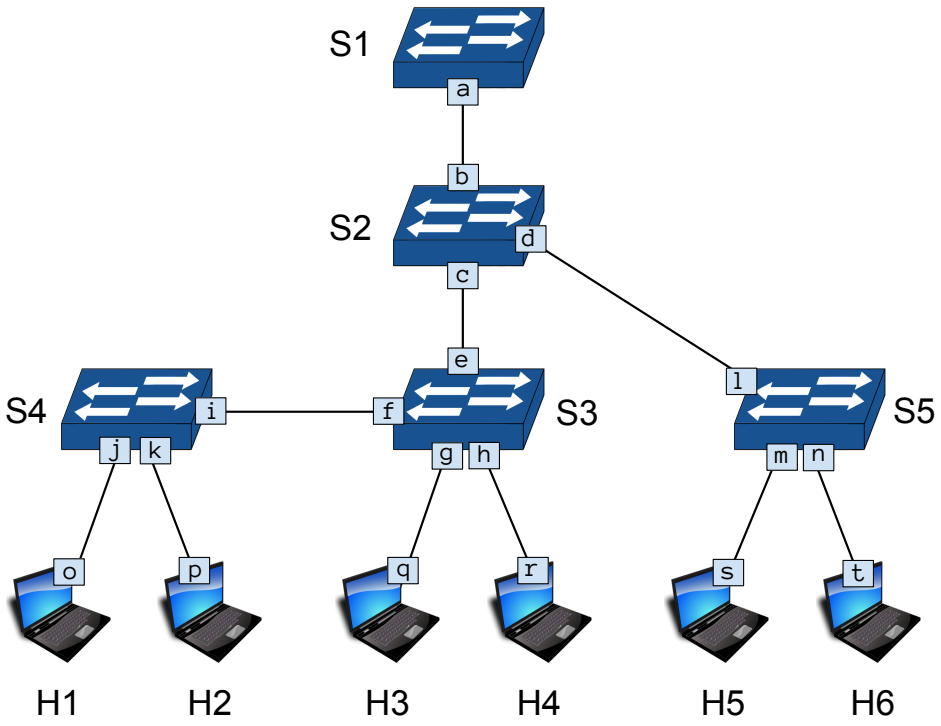

Figure 2: The network topology used in Problem A, Question 3

a. Describe the traffic generated in the network between the moment frame f_1 is sent by host $H1$ and the moment it is received by host $H6$: which devices send or receive which frame(s)?

Host $H1$ sends frame f_1 to switch $S4$.

Switch $S4$ does not have an entry in its forwarding table for the destination MAC address of the frame, so it forwards it to host $H2$, which ignores it, and to switch $S3$.

Switch $S3$ does not have an entry in its forwarding table for the destination MAC address of the frame, so it forwards it to host $H3$, which ignores it; to host $H4$, which also ignores it; and also to switch $S2$.

Switch $S2$ does not have an entry in its forwarding table for the destination MAC address of the frame, so it forwards to switch $S1$ and to switch $S5$.

Switch $S1$ does not have an entry in its forwarding table for the destination MAC address of the frame, but there is no other interface than the one on which the frame arrived (i.e. c), so it ignores it.

Switch $S5$ does not have an entry in its forwarding table for the destination MAC address of the frame, so it forwards it to host $H5$, which ignores it, and to host $H6$, which receives it.

b. Describe the traffic generated in the network between the moment frame f_2 is sent by host $H6$ and the moment it is received by host $H1$: which devices send or receive which frame(s)?

Host $H6$ sends frame f_2 to switch $S5$.

Switch $S5$ has an entry in its forwarding table for the destination MAC address of the frame, so it forwards it only to switch $S2$.

Switch $S2$ has an entry in its forwarding table for the destination MAC address of the frame, so it forwards it only to switch $S3$.

Switch $S3$ has an entry in its forwarding table for the destination MAC address of the frame, so it forwards it only to switch $S4$.

Switch $S4$ has an entry in its forwarding table for the destination MAC address of the frame, so it forwards it only to host $H1$.

c. Provide the entries in the forwarding tables of switches S_1 and S_2 after all the frames have finished traveling through the network. The entries must appear in the order they were generated. Fill in the answer in Tables 3 and 4.

MAC address	Outgoing port
Example: x	a
o	a

Table 3: Forwarding table of switch S_1

MAC address	Outgoing port
o	c
t	d

Table 4: Forwarding table of switch S_2

3 Problem B (30 points)

Question 1 (12 points):

Host A wants to communicate "securely" with Host B over TCP without using the complete suite of the SSL protocol. In each one of the following scenarios, they want to achieve a different purpose. Try to identify if there exists any flaw and provide:

i. An attack that exploits the flaw and defeats at least one of their purposes.

ii. A solution that fixes the flaw (i.e. what should A send instead).

Scenarios:

a. A wants to send a message m to B. It wants to ensure authenticity. For this, it relies on the fact that B knows A's IP address and can authenticate it. So, it sends message m to B.

b. A wants to send a message m to B. It wants to ensure authenticity and data integrity. For this, A sends $[m \mid H(m)]$, where H is a globally known cryptographic hash function.

c. A wants to send a sequence of messages m_i to B. It wants to ensure confidentiality, authenticity and all data integrity (i.e. B receives the same sequence of m_i sent by A). For this, for each m_i, A sends: $K_1 \{m_i \mid H(m_i|K_2)\}$, where K_1 and K_2 are two symmetric keys, shared between A and B.

d. A wants to send a sequence of messages m_i to B. It wants to ensure confidentiality, authenticity and all data integrity (i.e. B receives the same sequence of m_i sent by A). For this, for each m_i, A sends: $K_B^+ \left\{m_i \mid K_A^- \{H(m_i)\}\right\}$, where K_B^+ is B's public key, known to A; and K_A^- is A's private key.

*a. i. Authenticity and data integrity cannot be always guaranteed due to the **IP source spoofing** attack. It is not hard for an intruder (e.g. C) to create an IP datagram, put whatever IP source address (e.g. A's IP address) into the IP datagram, and send the datagram over the link-layer protocol to B IP spoofing.*

*ii. Instead of relying on the IP source address, A could send a **MAC** along with message m using one of the shared keys with B, i.e. A could send*

$$[m \mid H(m|K_1)] .$$

*Alternatively, A could **digitally sign** message m with its private key and send it to B, i.e it could send*

$$K_A^- \{H(m_i)\} .$$

*b. i. Authenticity cannot be guaranteed since the cryptographic hash function is **known to everyone**. Besides, data integrity is not assured: an intruder can intercept A's message, modify it to a new message m', compute its hash function $H(m')$, and send both of them to B, $[m' \mid H(m')]$. B can never detect such a man-in-the-middle attack and will wrongly think that message m' is A's message.*

*ii. As in (a), A could send a **MAC** along with the message: $[m \mid H(m|K_1)]$; or it could **digitally sign** it with its private key: $K_A^- \{H(m_i)\}$.*

c. i. *Even though confidentiality and authenticity can be guaranteed for every message because of the two symmetric keys that are properly used, it is not hard for an intruder to perform a message* ***insertion, deletion, reordering or replay attack*** *and violate data integrity of the entire sequence of* m_i*. E.g. he could capture at least two segments sent between* A *and* B*, reverse the order of the segments, adjust the TCP sequence numbers (not encrypted), and then send the two reverse ordered segments to* B*.*

ii. *A solution would be the use of* ***sequence numbers*** *inside the hash function. I.e. instead of sending* $K_1 \{m_i \mid H(m_i|K_2)\}$*,* A *could send*

$$K_1 \{m_i \mid H(m_i|K_2|\, SEQ)\}\,.$$

Both nodes should know that the first message sent has a specific sequence number (e.g. 0*) and the following have subsequent sequence numbers.* B *could track* A*'s sequence numbers, and upon an arrival of a message use them to evaluate the hash function and verify that there was no attack. TCP guarantees that all messages will arrive in order to* B *and thus such a solution can work, as an out-of-order message implies the existence of an attack.*

d. i. *The answer is similar to above. Even though confidentiality and authenticity can be guaranteed for every message because asymmetric key cryptography is properly used, it is not hard for an intruder to perform a message* ***insertion, deletion, reordering or replay attack*** *and violate data integrity of the entire sequence of* m_i*. E.g. he could capture at least two segments sent between* A *and* B*, reverse the order of the segments, adjust the TCP sequence numbers (not encrypted), and then send the two reverse ordered segments to* B*.*

ii. *A solution would be the use of* ***sequence numbers*** *inside the hash function. I.e. instead of sending* $K_B^+ \left\{m_i \mid K_A^- \{H(m_i)\}\right\}$*,* A *could send*

$$K_B^+ \left\{m_i \mid K_A^- \{H(m_i|SEQ)\}\right\}.$$

Both nodes should know that the first message sent has a specific sequence number (e.g. 0*) and the following have subsequent sequence numbers.* B *could track* A*'s sequence numbers, and upon an arrival of a message use them to evaluate the hash function and verify that there was no attack. TCP guarantees that all messages will arrive in order to* B *and thus such a solution can work, as an out-of-order message implies the existence of an attack.*

Question 2 (18 points):

Host A wants to secure its TCP connection to Web Server F with two symmetric keys K_a and K_c, one for authenticity and one for confidentiality, that are initially known only to A. For securing their communication, A and F use the following protocol:

- A sends a "hello" message.
- F sends a certificate containing its public key K_F^+.
- A sends $K_F^+ \{K_a | K_c\}$.
- A and F exchange any message m_i using both keys as: $K_c \{m_i \mid H(K_a | m_i)\}$, where H is a globally known cryptographic hash function.
- At the end of the session A and F exchange a MAC (Message Authentication Code) of all the exchanged messages. I.e. they send to each other: MAC$(m_1 \mid m_2 \mid m_3 \mid \ldots)$.

Answer the following:

a. Does this protocol guarantee that F gets the same sequence of messages m_i sent by A?
If yes, explain why. If not, describe an attack and give your solution.

Yes, because of two reasons:
(i) All messages m_i are send to B along with a MAC, for guaranteeing authenticity and message integrity, A and B are the only hosts that know the symmetric key K_a. As a result, upon a message arrival B can decrypt A's message (m_i), compute the MAC ($H(K_a | m_i)$), compare it to the MAC that was actually sent by A and finally authenticate A as the sender of each message.
(ii) A sends a MAC of all communicated messages (i.e. a hash of the concatenation of K_a and all m_i) to B in the end of the session. In this way, B can detect any insertion, deletion, reordering or replaying of messages and therefore be sure that the sequence of messages m_i that has received is the same as the one that A has sent.

b. Does this protocol guarantee that F gets the same sequence of messages sent by A only once?
If yes, explain why. If not, describe an attack and give your solution.

No, because a connection replay attack impersonating A is possible:

Trudy, a woman-in-the-middle can sniff and record all messages sent by A to F. Then, after the end of the session, she can resend all messages (one by one) to F, trying to impersonate A. Unfortunately, Web Server F will reply with exactly the same sequence of messages and will not detect any problem as the messages will pass the authenticity checks.

A simple solution would be to let F send a nonce (n) to A, along with its certificate. The nonce must then be used by A at every message sent to F. I.e. instead of sending $K_c \{m_i \mid H(K_a | m_i)\}$, A can send $K_c \{m_i \mid H(K_a | n | m_i)\}$. In this way, F (who knows n), can be sure that the sequence of the messages that it gets is sent by A only once and cannot be replayed.

[Other more complicated solutions, similar to the SSL approach, are also correct.]

c. Instead of using $K_c\{m \mid H(K_a|m)\}$, now A and F exchange their messages using $K_c\{m \mid H(K_a|m|N)\}$, where N is a number that increments each time A or F sends a message.

   i. Does this new protocol improve anything?

   ii. Is there any part of the new protocol that becomes unnecessary and why?

   i. Sending $K_c\{m \mid H(K_a|m|N)\}$ improves the initial protocol in the sense that: F can verify whether there exists (or not) a message insertion, deletion, reordering or replay attack, upon receiving each message, without having to wait until the end of the session to verify the data integrity of the entire sequence of messages. How? It can track what is the number N that A should use, and verify the authenticity and data integrity of the message by including N in the hash calculation. Since TCP guarantees that the messages arrive in order, when F receives an out-of-order message, this means that an intruder has inserted, deleted or reordered one or more messages.

   ii. The last part becomes now unnecessary, because the reason why it existed was to prevent an intruder form carrying out a man-in-the-middle attack such as inserting, deleting, reordering or replaying messages.

4 Problem C (30 points)

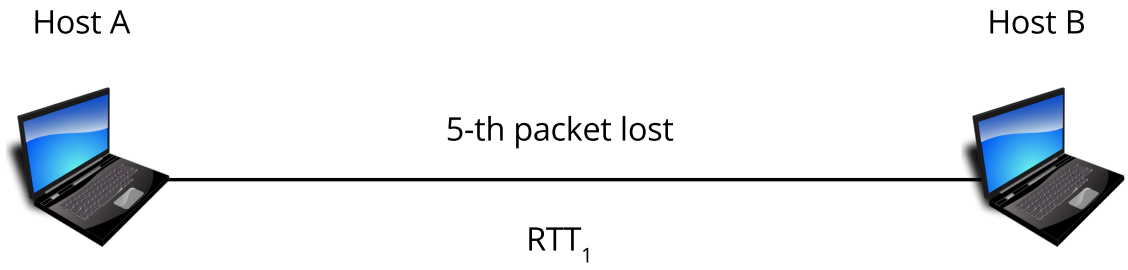

Figure 3: The network topology used in Problem C, Questions 1 and 2

Consider the network topology you see in Figure 3.

Host A opens a TCP connection to host B, and starts sending a file of size $F = 10$ bytes, in segments of size $MSS = 1$ byte each. As a result of a faulty link between A and B, the network drops the 5-th packet (without counting the SYN packet in the TCP handshake) transmitted by A.

For the following questions, make the following assumptions:

- The transmission delay for packets is negligible.
- The round-trip time between A and B is RTT_1.
- The sender timeout interval for each TCP flow is fixed, and equal to 2 times the round-trip time.
- TCP has Fast Retransmit disabled.
- A TCP receiver sends an ACK for each packet it receives.
- The first segment that A transmits will have a sequence number of 1.

Question 1 (10 points):

Complete the sequence diagram which shows:

- All packets exchanged between A and B.
- The sequence numbers sent by A and the ACK numbers sent by B.
- The phase that congestion algorithm is in (Slow Start, or Congestion Avoidance).
- The size of the congestion window, $cwnd$, of host A.
- The value of $ssthresh$ (the Slow Start threshold).

$cwnd_A$ [bytes]	$ssthresh_A$ [bytes]	State of the congestion control algorithm for host A
1	∞	—
1	∞	Slow start
2	∞	Slow start
3 4	∞	Slow start
5	∞	Slow start
1	2	Slow start
2	2	Congestion avoidance

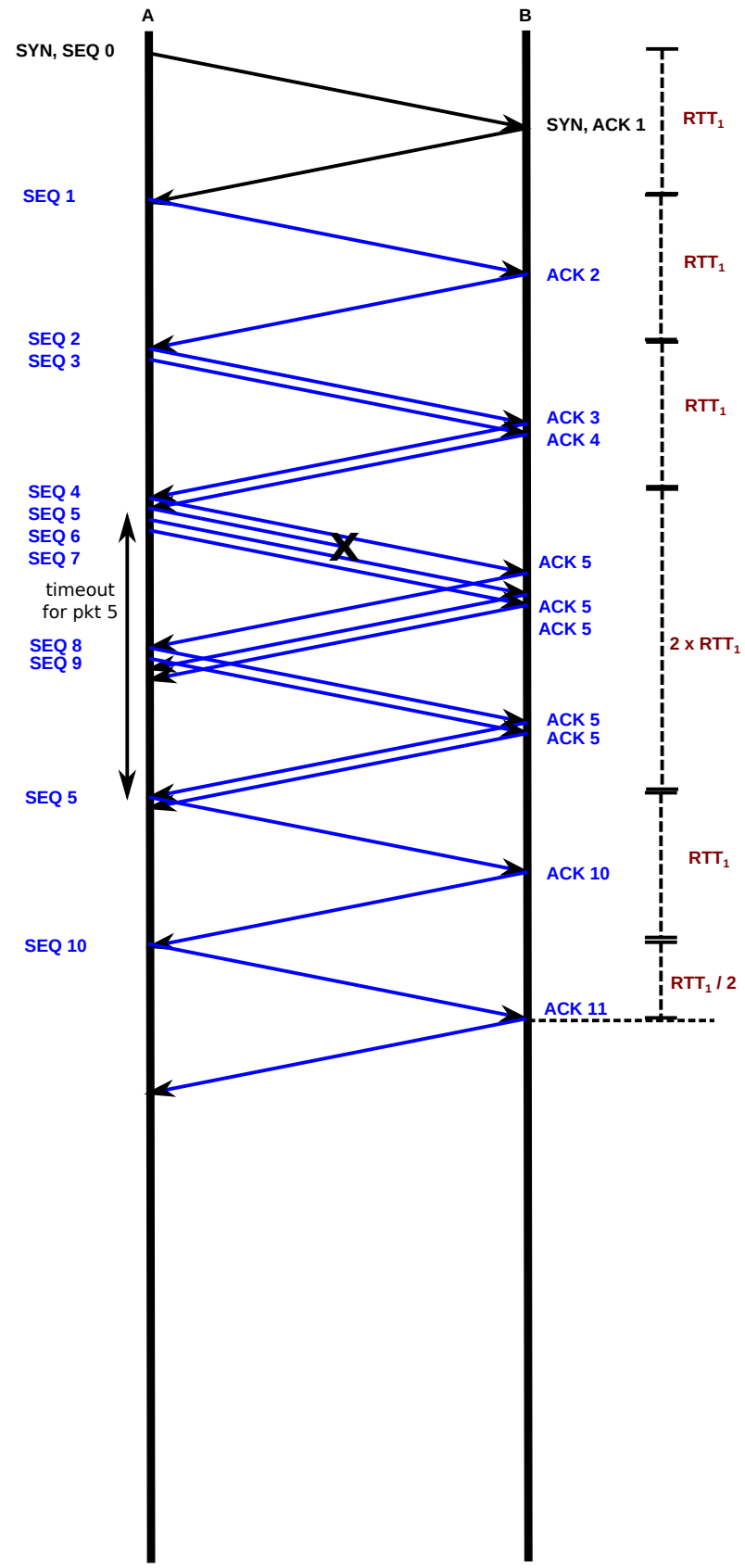

Figure 4: Sequence diagram to be completed for Question 1.

Question 2 (6 points):

For the sequence diagram you completed in the previous question, calculate how much time it takes for B to finish receiving the file.

(Note: The one-way propagation delay from A to B is $\frac{RTT_1}{2}$)

You can view the time durations marked down on the sequence diagram at figure **??**. *With the connection setup time included, it takes $6.5 \times RTT_1$ time to complete the file transfer, from start to finish.*

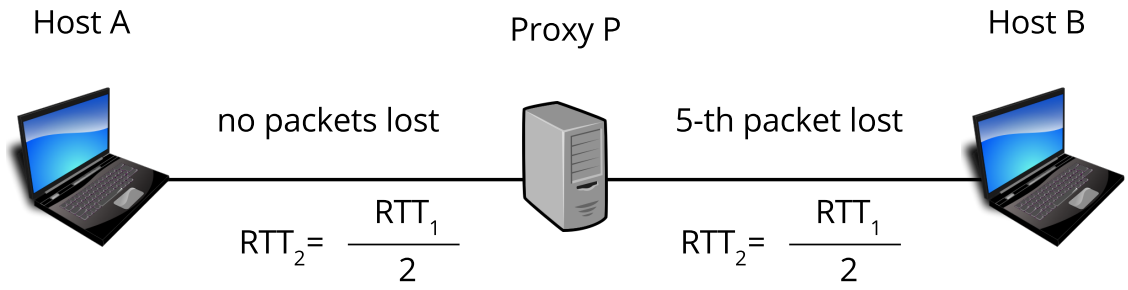

Figure 5: The network topology used in Problem C, Question 3

Question 3 (9 points):

Now, consider the network topology you see in Figure 5. A uses node P, which runs an application-layer proxy to transmit the file to B.

The proxy application receives data from a TCP socket connected to A (the input socket), and writes data out to a TCP socket connected to B (the output socket). P forwards these packets to the output socket, the moment it can read them from the input socket. The proxy's operations do not incur any processing delay.

P is located exactly in the middle of the path between A and B, such that the round-trip times between A and P, and between P and B are both equal to $RTT_2 = \frac{RTT_1}{2}$.

The faulty link described in the previous question is now located on the part of the path between P and B (the second half of the path). As a result, the 5-th packet transmitted on that part of the path is lost. No packet loss occurs on the part of the path between A and P.

Calculate the time it takes for the file transfer to be completed in this new setting.

(Note: Do not forget to adjust the timeout interval for the two TCP flows; from A to P, and from P to B. The timeout interval for the two flows is equal to $2 \times RTT_2 = RTT_1$)

The bottleneck will be at part of the path which contains the faulty link. For the first half of the path, we only need to consider the time it will take to propagate the first (connection) packet from A to P. This is equal to $\frac{RTT_2}{2}$.

From that point on, the part of the sequence diagram which will be on the second half of the connection will be an exact replica of the sequence diagram we created for Questions 1 and 2. The only difference now is that the round-trip time between P and B is half the round-trip time between A and B.

Thus, the total file transfer delay will be:

$$6.5 \times RTT_2 + \frac{RTT_2}{2} = 7 \times RTT_2 = \frac{7 \cdot RTT_1}{2}$$

Question 4 (5 points):

Does the introduction of the application-layer proxy in Question 3 improve or worsen the file transfer? Which features of TCP are responsible for this?

The main reason why the file transfer will take shorter to complete is the reduction of the end-to-end RTT for each flow. This reduction in RTT will have the following effects on TCP:

- *The congestion window will converge faster to an ideal value (assuming that link capacity is limited). In our example, where the capacity is unlimited, slow start will benefit even more.*
- *This will make TCP more responsive in detecting and correcting channel error. (e.g., the 5-th packet which was lost in Question 1). This is because the timeout interval adjusts to the RTT estimate; a smaller RTT means that a packet is retransmitted faster. We would observe the same behavior in the case where fast-retransmit was also activated.*