ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Computer Networks - Midterm

November 3, 2017

Duration: 2h15m

- This is a closed-book exam.
- Please write your answers on these sheets in a readable way, in English or in French.
- 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) (grades in red)

1 Short questions (10 points)

For each question, please circle a single best answer.

1. Caching of DNS responses can:
   (a) speed up hostname-to-IP address resolution.
   (b) save bandwidth.
   (c) improve resistance to flooding attacks.
   (d) All of the above. *(Correct)*

2. Compared to a packet switched network, a connection switched network:
   (a) has higher packet loss and delay.
   (b) does better use of bandwidth.
   (c) uses resources less efficiently. *(Correct)*
   (d) None of the above.

3. Which of the following is true about web cookies?
   (a) A cookie is state created by the web client.
   (b) Cookies speed up the delivery of web content.
   (c) Cookies allow web sites to tailor what the user sees on the screen. *(Correct)*
   (d) None of the above.

4. How many iterative DNS queries a local DNS server sends when trying to resolve `nal.epfl.ch`?
   (a) At most 1.
   (b) Exactly 3.
   (c) At most 3. *(Correct)*
   (d) More than 3.

5. Which of the following is true?
   (a) A socket enables a process to communicate with another process.
   (b) A process can use a UDP socket to communicate with many remote processes.
   (c) A process can use a TCP socket to communicate with only one remote process.
   (d) All of the above. *(Correct)*

6. A Distributed Hash Table (DHT) stores:

   (a) content files.

   (b) only metadata files.

   (c) only popular content.

   (d) the identities of the peers that store each content file. *(Correct)*

7. Suppose that we double the rate of all links in a network. Which of the following is true?

   (a) The end-to-end delay will decrease by at most 50%. *(Correct)*

   (b) The end-to-end delay will decrease exactly by 50%.

   (c) The propagation delay will decrease exactly by 50%.

   (d) The queuing delay will decrease exactly by 50%.

8. A user generates data as follows: 1% of the time he is active and produces data at 100 Mbps, while he is silent for the rest of the time. How many such users can a link of 1 Gbps support while guaranteeing no packet loss?

   (a) Less than 10.

   (b) Exactly 10. *(Correct)*

   (c) More than 10.

   (d) We cannot say precisely, all the above can be true.

9. For a 100 Kbps link with a 40 msec end-to-end propagation delay and packets of 8 Kbits, what is the sender utilization (the fraction of time the sender is busy), if the Stop-and-wait protocol is used and the size of the ACKs is 1 bit?

   (a) Less than 0.5. *(Correct)*

   (b) Exactly 0.5.

   (c) More than 0.5.

   (d) We cannot say precisely, all the above can be true.

10. Suppose a switch with an initially empty buffer of infinite capacity and an outgoing link with transmission rate 1Mbps. Each packet is of 100Kbits and batches of 10 of them simultaneously arrive at the switch every 1 sec. The average queuing delay a packet experiences is therefore:

    (a) 0.1 sec.

    (b) between 0.1 and 1 sec. *(Correct)*

    (c) 1 sec.

    (d) None of the above.

2 Web Browsing and DNS (30 points)

Setup:

Three users, Alice, Bob, and Persa, are logged into their workstations, respectively called `alice.ethz.ch`, `bob.ethz.ch`, and `persa.ethz.ch`, all located inside ETHZ's network.

ETHZ offers a web server `www.ethz.ch` and a local name server `ns.ethz.ch`, which is also the authoritative server for the `ethz.ch` domain.

EPFL offers a web server `www.epfl.ch` and a local name server `ns.epfl.ch`, which is also the authoritative server for the `epfl.ch` domain.

The setup is illustrated in Figure 1. Make the following assumptions:

- All DNS caches are initially empty.
- All DNS requests are resolved recursively.
- Each HTTP message fits in a single packet.
- The web browsers and servers use persistent HTTP connections.

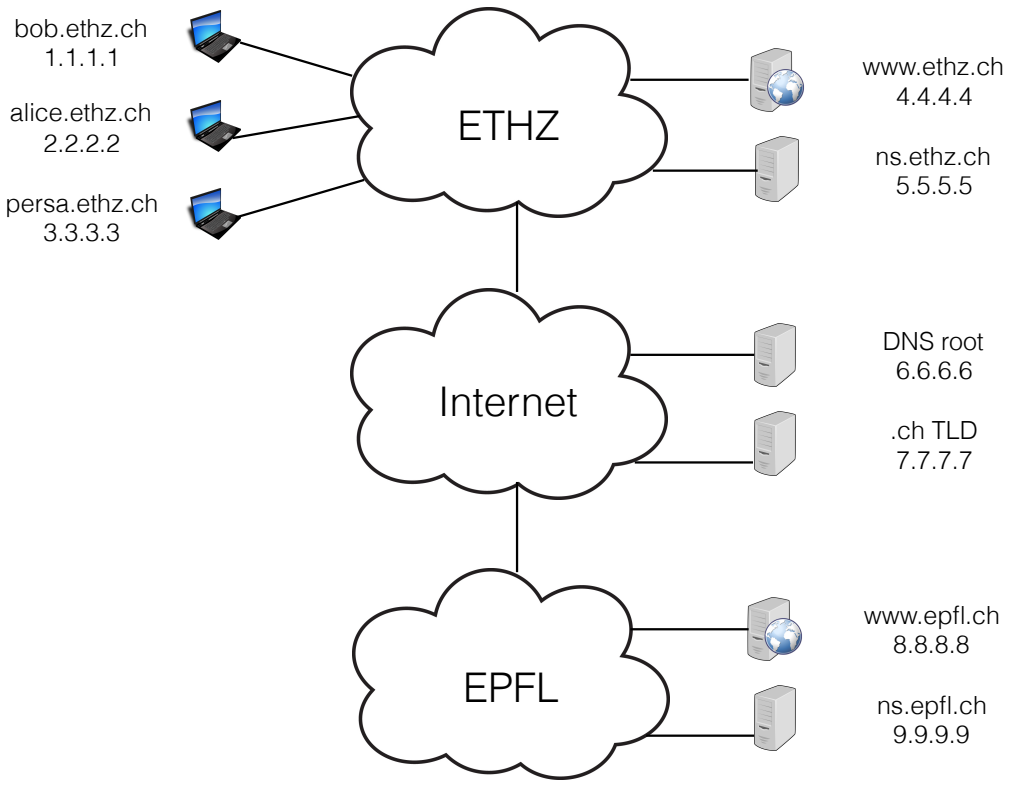

Figure 1: The setup for the exercise on Web browsing and DNS

Question 1 (10 points):

Alice uses her web browser to access URL `http://www.epfl.ch/index.html`. This page references one image with URL `http://www.epfl.ch/image.jpg` and another web page with URL `http://www.ethz.ch/file.html` (which does not reference any other object).

List all packets that are exchanged in the entire network, including any connection-setup packets, by filling Table 1. For each packet, show the source and destination IP address, the transport-layer protocol, the application-layer protocol, and the purpose of the packet, as in the example.

Packet	Source IP	Dest. IP	Transport protocol	Application protocol	Purpose
ex.1	1.0.0.1	1.0.0.2		HTTP	HTTP reply with `image.jpg`
1	2.2.2.2	5.5.5.5	UDP	DNS	www.epfl.ch IP address?
2	5.5.5.5	6.6.6.6	UDP	DNS	www.epfl.ch IP address?
3	6.6.6.6	7.7.7.7	UDP	DNS	www.epfl.ch IP address?
4	7.7.7.7	9.9.9.9	UDP	DNS	www.epfl.ch IP address?
5	9.9.9.9	7.7.7.7	UDP	DNS	www.epfl.ch: 8.8.8.8
6	7.7.7.7	6.6.6.6	UDP	DNS	www.epfl.ch: 8.8.8.8
7	6.6.6.6	5.5.5.5	UDP	DNS	www.epfl.ch: 8.8.8.8
8	5.5.5.5	2.2.2.2	UDP	DNS	www.epfl.ch: 8.8.8.8
9	2.2.2.2	8.8.8.8	TCP		Connection request
10	8.8.8.8	2.2.2.2	TCP		Connection response
11	2.2.2.2	8.8.8.8	TCP	HTTP	GET request for index.html
12	8.8.8.8	2.2.2.2	TCP	HTTP	index.html
13	2.2.2.2	8.8.8.8	TCP	HTTP	GET request for image.jpg
14	8.8.8.8	2.2.2.2	TCP	HTTP	image.jpg
15	2.2.2.2	5.5.5.5	UDP	DNS	www.ethz.ch IP address?
16	5.5.5.5	2.2.2.2	UDP	DNS	www.ethz.ch: 4.4.4.4
17	2.2.2.2	4.4.4.4	TCP		Connection request
18	4.4.4.4	2.2.2.2	TCP		Connection response
19	2.2.2.2	4.4.4.4	TCP	HTTP	GET request for file.html
20	4.4.4.4	2.2.2.2	TCP	HTTP	file.html

Table 1: Packets transmitted on the Internet, in Question 1

Question 2 (5 points):

Which is the minimum number of sockets on Alice's machine that are necessary for the scenario in Question 1? What kind of sockets are they? Which of the packets that you listed in Table 1 are sent and received over each socket?

Three sockets:

- *1 UDP socket for communication with ns.ethz.ch: packets 1, 8 , 15, 16*
- *1 TCP socket for communication with www.epfl.ch: packets 9 – 14*
- *1 TCP socket for communication with www.ethz.ch: packets 17 – 20*

Question 3 (5 points):

After Alice has viewed `http://www.epfl.ch/index.html`, Bob wants to access the same URL.

Persa is a malicious user who guesses exactly when Bob tries to access `http://www.epfl.ch/index.html`. She wants to trick Bob and make him access a web server running on her own workstation, thinking that he is accessing the EPFL web server.

How can Persa do that by sending DNS traffic to Bob?

When Bob's DNS client makes a DNS request to `ns.ethz.ch` for `www.epfl.ch`'s IP address, Persa can impersonate `ns.ethz.ch` and respond that `www.epfl.ch`'s IP address is `3.3.3.3` (Persa's IP address). If Persa's response gets to Bob's DNS client before the real `ns.ethz.ch`'s response, Bob's web browser will connect to the web server running on Persa's workstation instead of `www.epfl.ch`.

Question 4 (10 points):

Describe in more detail Persa's attack from Question 3:

a) Describe the packet(s) sent by Persa. What will be the source IP address and port number, destination IP address and port number, transport-layer protocol, and application-layer protocol of each packet? (6 points)

This is a relatively hard question. Unless you have already thought about this attack and know the answer, we suggest leaving it for the end.

Persa must trick Bob's DNS client into accepting her DNS response as if it came from `ns.ethz.ch`*. The challenge is that Persa does not know the port number of the UDP socket where Bob's DNS client is waiting to receive* `ns.ethz.ch`*'s response. Hence, Persa needs to send as many packets as there are port numbers, each packet with a different destination port number, such that one of them has the right destination port number and is accepted by Bob's DNS client.*

Persa's packets will have the following fields:

- *Source IP address: 5.5.5.5 (*`ns.ethz.ch`*'s IP address)*
- *Destination IP address: 1.1.1.1 (Bob's IP address)*
- *Source port number: 53 (the standard DNS port number)*
- *Destination port number: each packet a different one, covering the entire range port numbers*

b) List all the packets that are sent by all the other machines in the network, including any connection-setup packets, by filling Table 2. For each packet, show the source IP address, destination IP address, transport-layer protocol, application-layer protocol and its purpose. (4 points)

Packet	Source IP	Dest. IP	Transport protocol	Application protocol	Purpose
ex.1	`1.0.0.1`	`1.0.0.2`		HTTP	HTTP reply with `image.jpg`
1	`1.1.1.1`	`5.5.5.5`	UDP	DNS	www.epfl.ch's IP address?
2	`5.5.5.5`	`1.1.1.1`	UDP	DNS	www.epfl.ch: 8.8.8.8
3	`1.1.1.1`	`3.3.3.3`	TCP		Connection request
4*	`3.3.3.3`	`1.1.1.1`	TCP		Connection response
5	`1.1.1.1`	`3.3.3.3`	TCP	HTTP	GET request for index.html
6*	`3.3.3.3`	`1.1.1.1`	TCP	HTTP	index.html
7**	`1.1.1.1`	`3.3.3.3`	TCP	HTTP	GET request for image.jpg
8**	`3.3.3.3`	`1.1.1.1`	TCP	HTTP	image.jpg
9**	`1.1.1.1`	`5.5.5.5`	UDP	DNS	www.ethz.ch's IP address?
10**	`5.5.5.5`	`1.1.1.1`	UDP	DNS	www.ethz.ch: 4.4.4.4
11**	`1.1.1.1`	`4.4.4.4`	TCP		Connection request
12**	`4.4.4.4`	`1.1.1.1`	TCP		Connection response
13**	`1.1.1.1`	`4.4.4.4`	TCP	HTTP	GET request for file.html
14**	`4.4.4.4`	`1.1.1.1`	TCP	HTTP	file.html

Table 2: Packets transmitted on the Internet, in Question 4

* Packets 4, 6, and 8 are not a necessary part of the answer, because the question asks to list all packets sent by end-systems *other than Persa*. We have included them for clarity.

** Packets 8 – 14 are not a necessary part of the answer, because Persa's index.html may not have referenced image.jpg and file.html. It is correct both to include them and to not include them.

3 Reliable Data Transport (30 points)

Question 1 (6 points):

End-host A wants to send a file to end-host B using a transport-layer protocol that provides reliable data delivery with pipelining and sender window N=5. A's transport layer splits the file in 10 segments with sequence numbers from 0 to 9. **The first data segment from A to B is lost the first time it is transmitted**, while all other transmissions arrive at their destination uncorrupted and in order.

Complete the two diagrams in Figure 2 to show all the segments (with sequence numbers and ACK numbers) transmitted by A and B, in each of the following two cases:

- A and B use Go-Back-N (left part of Figure 2).
- A and B use Selective Repeat (right part of Figure 2).

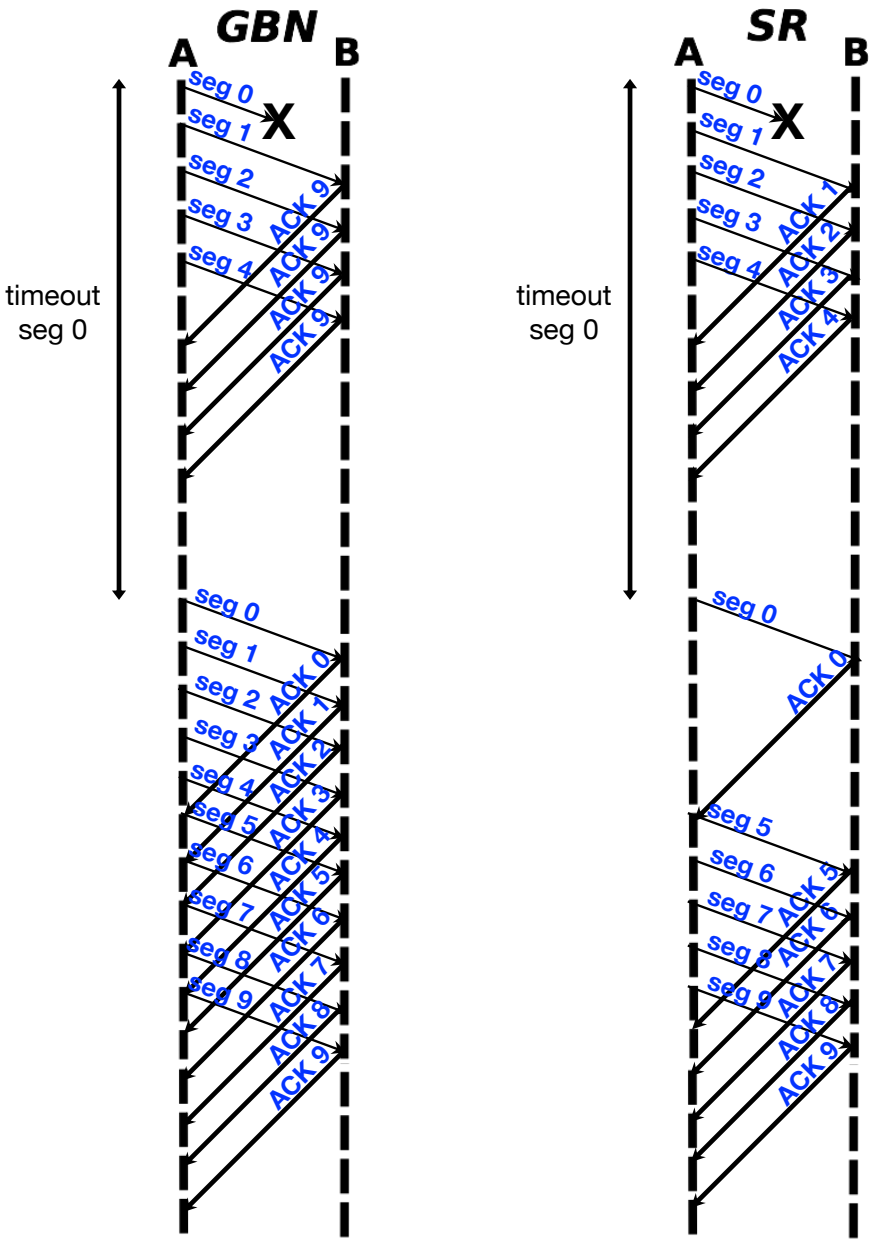

Figure 2: Solution to Question 1.

Question 2 (6 points):
Consider the same scenario as in Question 1, with the following difference: **The first 4 ACK segments from B to A are lost**, while all other segments arrive at their destination uncorrupted and in order.

Complete the two diagrams in Figure 3 to show all the segments (with sequence numbers and ACK numbers) transmitted by A and B, in each of the following two cases:

- A and B use Go-Back-N (left part of Figure 3).
- A and B use Selective Repeat (right part of Figure 3).

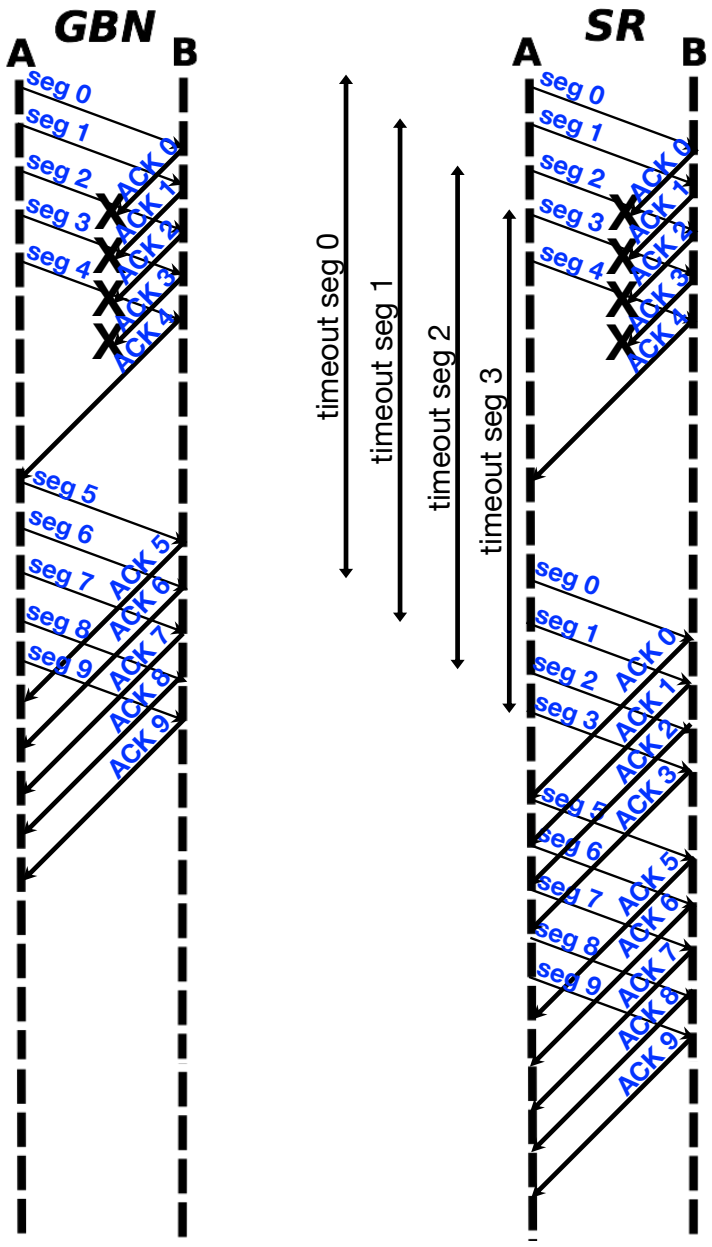

Figure 3: Solution to Question 2.

Question 3 (3 points):
Based on your answers to Questions 1 and 2, can you draw a general conclusion about when one should prefer Go-Back-N over Selective Repeat, and when one should prefer Selective Repeat over Go-Back-N? Justify your answer, e.g., if you state that one should prefer Go-Back-N in situation X, please say what metric Go-Back-N improves relative to Selective Repeat in situation X.

Selective Repeat is expected to achieve higher sender utilization and throughput than Go-Back-N when there are few, independent packet losses.

Go-Back-N is expected to achieve higher sender utilization and throughput than Selective Repeat when there is bursty loss on the reverse channel (many ACKs get lost back to back).

Question 4 (15 points):
In class, we discussed different elements of reliable data transfer (RDT): checksums, acknowledgments, timeouts, sequence numbers, and pipelining.

End-host A wants to send data reliably to end-host B (B never sends data to A). Which of the above RDT elements do they need to use in each of the following scenarios and why? In all scenarios except for (e), A wants to send data to B with high throughput.

a) The path from A to B may corrupt packets, but never drops or reorders packets. The path from B to A never corrupts, drops or reorders packets. There are no queuing/processing delays on either path.

We need:

- *checksums to detect the corruption,*
- *ACKs to signal to the sender that a packet was (not) corrupted,*
- *pipelining for high throughput.*

We don't need:

- *timeouts, because there is no loss or unexpected delay on the reverse path, hence the sender always receives every positive or negative ACK sent by the receiver;*
- *sequence numbers, because there is no loss or reordering anywhere and no corruption on the reverse path, hence the receiver always knows which segment it is receiving (and whether it's a retransmission), and the sender always knows which segment is being ACK-ed.*

b) Both paths (from A to B, and from B to A) may corrupt packets, but they never drop or reorder packets, and they never introduce queuing/processing delays.

We need:

- *checksums to detect the corruption,*
- *ACKs to signal to the sender that a packet was (not) corrupted,*
- *sequence numbers, so that the receiver knows whether is its receiving a new segment or a retransmission,*
- *pipelining for high throughput.*

We don't need:

- *timeouts, because there is no loss or unexpected delay on the reverse path, hence the sender always receives every positive or negative ACK sent by the receiver.*

c) Both paths (from A to B, and from B to A) may drop or reorder packets or introduce queuing/processing delays, but they never corrupt packets.

We need:

- *ACKs and timeouts to detect that a packet was lost or unpredictably delayed,*
- *sequence numbers, so that the sender knows which packet is being ACK-ed by the receiver,*
- *pipelining for high throughput.*

We don't need:

- *checksums, because there is no corruption.*

d) The path from A to B never corrupts, drops or reorders packets, and never introduces queuing/processing delays. The path from B to A may do any of these things.

We need no RDT elements, because the path from A to B is perfect.

e) A sends one small piece of data (small enough to fit in one segment) to B every hour. Both paths (from A to B, and from B to A) may corrupt, drop or reorder packets or introduce queuing/processing delays.

We need all RDT elements except for pipelining, which is not needed, because A does not need to send data to B at a high throughput.

4 Delay Computation (30 points)

The network in Figure 4 contains the following elements:

- Server S, which stores a file.
- N "odd" nodes and N "even" nodes, where $N \geq 2$.
- Packet switches $S1$, $S2$ and $S3$, which perform store-and-forward packet switching and introduce insignificant processing delays. Each switch has a separate (infinite) queue for each outgoing link. E.g., switch S_2 has three separate queues: one queue for packets addressed to the "odd" nodes, one queue for packets addressed to the "even" nodes, and one queue for packets addressed to server S. As soon as S_2 receives a packet, it immediately places the packet in one of the three queues, depending on the packet's destination.

Assume that:

- Each link has length ℓ and propagation speed c.
- The transmission rate of a link is the same in both directions of the link.
- The end-systems communicate over UDP. When you compute delays, you do not need to worry about connection setup.
- The end-systems communicate using the client-server architecture. There is no P2P file distribution in this problem.
- No other approximation/simplifying assumption can be made.

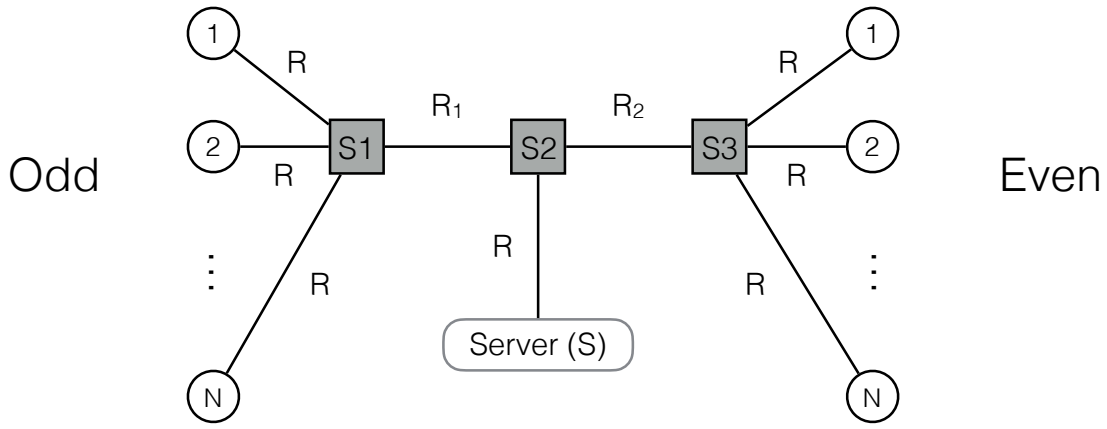

Figure 4: Network topology

Question 1 (15 points): Suppose that: $R_1 < R_2 < R$.

a. Each "odd" node sends a request to server S for the file (no "even" node sends a request). All the "odd" nodes start transmitting their requests simultaneously. Each request fits in a single packet of size Q bits. How long does it take for all the "odd" requests to reach the server S? (7 points)

$$d_{total} = \frac{Q}{R} + N\frac{Q}{R_1} + \frac{Q}{R} + 3\frac{\ell}{c}$$
$$= \frac{2Q}{R} + N\frac{Q}{R_1} + 3\frac{\ell}{c}$$

b. Each "odd" and each "even" node has sent a request to server S for the file, and S has received all $2N$ requests. S splits the file into P packets of equal length L and sends a copy of the file to each node. S sends the file first to all the "even" and then to all the "odd" nodes.

How long does it take for the file to be fully delivered to all the "odd" and "even" nodes? (8 points)

$d_{total} =$ *time for transmitting all data for the even nodes over the link:* $S \rightarrow S_2$
+ time for transmitting the first packet to the first odd node over the link: $S \rightarrow S_2$
+ time for transmitting all data for the odd nodes over the link: $S_2 \rightarrow S_1$
+ time for transmitting the last packet to the last odd node over the last link
+ propagation delay

$$= NP\frac{L}{R} + \frac{L}{R} + NP\frac{L}{R_1} + \frac{L}{R} + 3\frac{\ell}{c}$$

Question 2 (15 points): Now, suppose that: $R < R_1 < R_2$ and $NR_1 > R_2$.

a. Each "odd" and each "even" node sends a request for the file to server S. All nodes start transmitting their requests simultaneously. Each request fits in a single packet of size Q bits.

(i) Compute the time $T_{first.odd}$ at which the first "odd" request arrives at S_2. (1 point)

(ii) Compute the time $T_{last.even}$ at which the last "even" request arrives at S_2. (1 point)

(iii) Will the first "odd" request experience queuing delay at switch S_2? (2 points)

(iv) How long does it take in total for all the "odd" and "even" requests to reach S? (3 points)

(i) See next answer.

(ii) Let:

- $T_{first.odd}$ *be the time at which the first odd request arrives at* S_2,
- $T_{last.odd}$ *be the time at which the last odd request arrives at* S_2,
- $T_{first.even}$ *be the time at which the first even request arrives at* S_2,
- $T_{last.even}$ *be the time at which the last even request arrives at* S_2.

Then:

$$T_{first.odd} = \frac{Q}{R} + \frac{Q}{R_1} + 2\frac{\ell}{c}$$
$$T_{last.odd} = \frac{Q}{R} + N\frac{Q}{R_1} + 2\frac{\ell}{c}$$
$$T_{first.even} = \frac{Q}{R} + \frac{Q}{R_2} + 2\frac{\ell}{c}$$
$$T_{last.even} = \frac{Q}{R} + N\frac{Q}{R_2} + 2\frac{\ell}{c}$$

(iii) Since $R_1 < R_2 < NR_1$, we get:

$$T_{first.even} < T_{first.odd} < T_{last.even} < T_{last.odd} \quad (1)$$

All above inequalities hold because $R_1 < R_2$, except for the inequality $T_{first.odd} < T_{last.even}$, which holds because $R_2 < NR_1 \Leftrightarrow \frac{1}{R_1} < N\frac{1}{R_2}$.
Ineq. 1 says that the odd and even requests will arrive at S_2 in a mixed order, and that the first request to arrive will be even, while the last one to arrive will be odd. Moreover, since $R < R_1 < R_2$, the bottleneck of the path is the link $S_2 \rightarrow S$. Therefore all the requests will queue at the buffer of the outgoing link of S_2 towards S and there will be no time "gap" during which S_2 has no request to transmit to S.
Another solution could be to prove that the departure time from S_2 of an even packet that arrives before the first odd packet (e.g. the first even packet), is larger than the arrival time of the first odd packet at S_2. This signifies that the first odd packet will experience queuing delay at S_2.
Both solutions were considered correct.

(iv)

$$\begin{aligned} d_{total} = & \quad \textit{time for transmitting the first even request over the link: } Node \rightarrow S_3 \\ & + \textit{time for transmitting the first even request over the link: } S_3 \rightarrow S_2 \\ & + \textit{time for transmitting all requests over the link: } S_2 \rightarrow S \\ & + \textit{propagation delay} \\ = & \quad \frac{Q}{R} + \frac{Q}{R_2} + 2N\frac{Q}{R} + 3\frac{\ell}{c} \end{aligned}$$

b. Server S has received all $2N$ requests sent in Question 2b. S splits the file into P packets of equal size L and sends a copy to each node. S sends the file first to all the "even" nodes and then to all the "odd" nodes.

How long does it take for the file to be fully delivered to all the "odd" and "even" nodes? (8 points)

Note that since the queues of each outgoing link of S_1 are separate and since the links $S \rightarrow S_2$ and $S_1 \rightarrow$ Each-Odd-Node have the same transmission rate, which also happens to be the lowest transmission rate of the path, there will be no additional queuing delays S_1: By the time that the i-th packet for node j arrives at S_1, the $i-1$-th packet for the same node j has already departed from the j-th queue.

Hence, we can answer this question by considering the bottleneck link to be link $S \rightarrow S_2$.

$$
\begin{aligned}
d_{total} = \quad & \textit{time for transmitting all data for the even nodes over the link: } S \rightarrow S_2 \\
& + \textit{time for transmitting all data for the odd nodes over the link: } S \rightarrow S_2 \\
& + \textit{time for transmitting the last packet to the last odd node over the link: } S_2 \rightarrow S_1 \\
& + \textit{time for transmitting the last packet to the last odd node over the last link} \\
& + \textit{propagation delay} \\
= \quad & 2NP\frac{L}{R} + \frac{L}{R_1} + \frac{L}{R} + 3\frac{\ell}{c}
\end{aligned}
$$