Computer Communication
Exam: EDA343, DIT423; Re-exam EDA344, LEU062

Time and Place: Tuesday 31/05, 2022, 08.30-12.30

Course Responsible: Romaric Duvignau (EDA343, DIT423, LEU062), Marina Papatriantafilou (EDA344, DIT423) (Tel: 031 772 6976, 031 772 5413)

Grading:
CTH - EDA344, EDA343, LEU062: 30-40, 41-50, 51-60 → 3, 4, 5
GU - DIT423: 30-45, 46-60 → G, VG

Allowed material:

- English-X (X can be French, German, Swedish, etc) dictionary
- *No other books, no notes, no calculators, no electronic devices.*

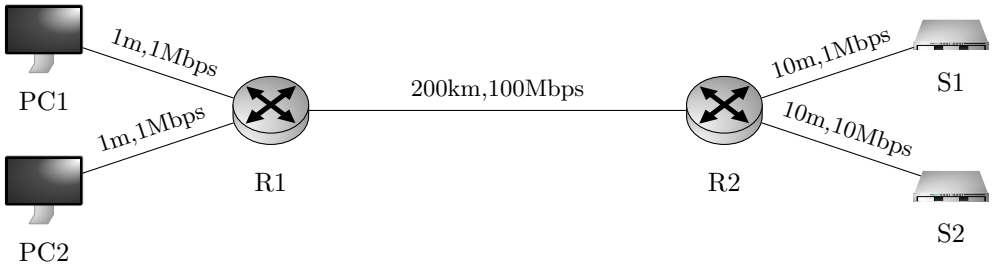
Instructions

- **Write clearly your course-code (EDA343/DIT423/LEU062/EDA344)**
- **Start answering each assignment on a new page; use only one side of each sheet of paper; please sort the sheets according to the question-ordering and number them.**
- Write in a **clear manner** and **motivate** (explain, justify) your answers. If it is not clear what is written for some answer, it will be considered wrong. If some answer is not explained/justified, it will get **significantly** lower marking.
- If you make any **assumptions** in answering any item, do not forget to clearly state what you assume.
- A good **rule-of-thumb for the extent of detail to provide**, is to include enough information/explanation so that a person, whose knowledge on computer communication is at the level of our introductory lecture, can understand.
- Please answer in English, if possible. If you have large difficulty with that (with all or some of the questions) and you think that your grade might be affected, feel-free to write in Swedish.
- Inspection of exam: date and time will be announced on the front page of the course in the canvas system.

Good Luck !!! Lycka till !!!!

1. **Computer Networks and the Internet (10pts)**

Consider the following network, made of 2 computers (PC1, PC2), 2 servers (S1,S2) and 2 routers (R1,R2):



Above each link of the network is mentioned the values "D,B" where D is the length of the link and B is the bandwidth of the link. In this exercise, we will assume that:

- All ping packets are exactly **1000 bits** long.
- Propagation speed is exactly **200km/ms**.
- Nodal processing time is exactly **1ms**.
- Queuing delay is **0ms**.
- Bandwidth is shared equally among concurrent TCP connections.
- As usual, 1 Mbps = 10^6 bits per second = 1000 bits per ms.

(a) (4pts) Under our assumptions, calculate the latency returned by running from PC1 (i) `ping PC2` and (ii) `ping S2`. You can round your answer to closest millisecond.

(b) (3pts) Suppose that using HTTP, PC1 downloads a very large file stored on the server S1. What will be the average throughput of the download assuming (i) this is the only traffic on the network and (ii) PC2 is simultaneously downloading a very large file from S2.

(c) (3pts) The Internet is using a packet-switching approach for handling communication. There is an alternative approach called "circuit switching" that does not rely on *packets* in order to communicate. What are the pros and cons of using packet switching versus circuit switching? (Feel free to use a table to organize your answer).

Answers:

(a) (i) transmitting a packet on a 1Mbps link takes (packet-size)/(bandwidth) = 1ms, the router needs 1 ms to switch the packet, hence in total it will take 2*(1+1+1) = 6ms for a ping to come back (propagation delay can be neglected as $\ll$ 1ms). (ii) For propagation delay, only the central link R1-R2 is not neglected and entails 1ms delay, hence RTT will be here 2 * (1+1+1+0.01+1+0.1) = 8.22ms = 8ms.

(b) In both cases, the bottleneck link is the link PC1-R1 (similar to PC2-R1) limiting the connection to 1Mbps as the shared link R1-R2 will provide 50Mbps each.

(c)

Approach	Pros	Cons
Packet-switching	Better for bursty data; simpler to implement; more efficient.	Packet losses; no guarantees.
Circuit-switching	No packet loss; Can provide bandwidth guarantees.	Resources can be wasted; More complicated.

2. **Application Layer (10pts)**

In this exercise, we suppose a client PC1 opens the webpage `https://www.cineflix.se/` and retrieves a URL to a video resource pointing to `https://video.cineflix.se/abcd`. Cineflix is a streaming service that uses DASH protocol to stream the videos to the clients and uses the CDN provider amassonCDN to host all its video content. As often, we assume that amassonCDN uses DNS to intercept and redirect the client requests, that is Cineflix will be sending an alias upon receiving a request for `video.cineflix.se`.

(a) (5pts) Reproduce and complete the following figure illustrating the process on how PC1 gets to know using its Local DNS (LDNS) a CDN server that will provide it with the requested content:

On the figure, `dns.cineflix.se` is an authoritative name server on `cineflix.se` domain, `dns.amassonCDN.se` is an authoritative name server on `amassonCDN.se` domain, `dns.se` is a TLD name server for `.se` TLD. We assume both `www.cineflix.se`, `dns.cineflix.se` and `dns.se`'s IP addresses are already cached on LDNS (and nothing else) and PC1 has already retrieved `www.cineflix.se`'s IP address. Also, the LDNS resolves DNS queries in an iterative fashion.

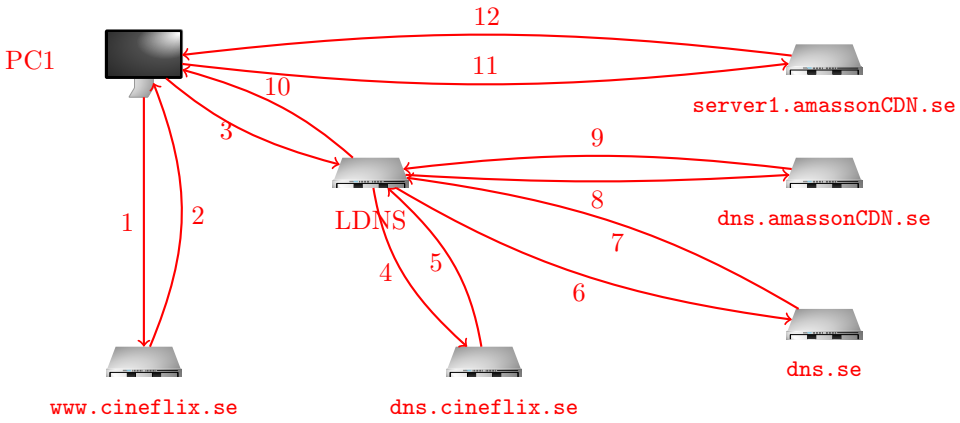
You should add to the figure the different communication by using **numbered directed edges** (where the number indicates the order of the interaction). The first edge is given and corresponds to the HTTP GET request from PC1.

List all the edges you added and for each give a short explanation for what it corresponds to, eg "HTTP GET request" for (1).

(b) (2pts) What information can amassonCDN use to decide which CDN server to return to the client?

(c) (3pts) What are the advantages for DASH to use HTTPS and thus TCP/TLS compared with using UDP for better efficiency?

Answers:



(a)

1. HTTP GET for the webpage
2. HTTP Response: 200 OK, containing the link to the video
3. DNS type A query for `video.cineflix.se`
4. LDNS forward the DNS query to `dns.cineflix.se`
5. `dns.cineflix.se` replies with an alias, eg `server1.amassonCDN.se`
6. DNS type NS query for domain `amassonCDN.se` sent to `dns.se`
7. DNS reply with name and IP for `dns.amassonCDN.se`
8. DNS type A query for `server1.amassonCDN.se`
9. DNS reply with a content server's IP for `server1.amassonCDN.se`
10. LDNS forward the DNS reply to the host

11-12+. host establishes a TCP connection with `server1.amassonCDN.se`, which one will send video segments following the DASH protocol (first retrieving of a manifest file, etc)

(also OK if PC1 is resolving the initial query in a iterative fashion).

(b) The only information that is readily available for the CDN provider to choose a content server is the IP of the LDNS, which can be mapped to a geographical position using geo-localisation databases for IPs. Once with a geographical position, the CDN provider can pick one of its geographically close clusters. This approach will fail if the client is configured to use a remotely located LDNS (eg Google's public DNS "8.8.8.8") and here DNS alone is not helpful for cluster selection.

(c) There are several advantages that are combined in HTTPS versus UDP, namely:

- Security. HTTPS provides encryption hence the video traffic is encrypted and thus not visible to potential eavesdropper.
- RDT. HTTPS relies on TCP (with TLS encryption) which includes reliable data transfer, congestion control, flow control etc which will be required to implement at the application layer if not included. DASH provides a bandwidth adaptation mechanism which could be used to adapt UDP sending rate but we will need packet-level encryption if we want to keep the "secure" aspect.

Both advantages can be recovered by using HTTP3/QUIC which runs over UDP.

3. **Transport Layer (10pts)**
Let's consider in this exercise packets of fixed length of exactly **L = 1100 bytes**, acknowledgments of negligible size (**0 bytes** to simplify) and that a client C is communicating with a remote server S crossing several routers. We further assume that the access link of the client which has a bandwidth of **R = 1100 bytes/ms** is the bottleneck link, that there are no packet losses and that the RTT between the client and the server is constant and equals to **10ms**. For TCP, we assume **MSS = L** and **ssthresh = 8** MSS.

(a) (3pts) Suppose a "stop-and-wait" protocol is used (i.e., waiting for an acknowledgment before proceeding to next packet). What is then the maximum sending rate between the client and the server? (Illustrate your answer by a simple time-diagram; your answer can be in bytes/ms or KB/s).

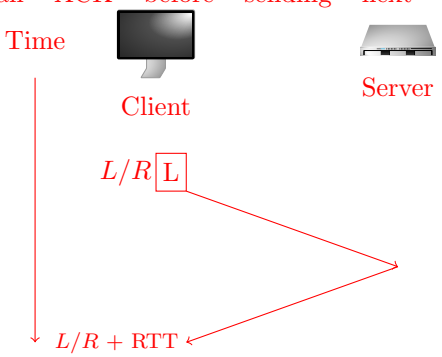
(b) (4pts) Suppose now we use TCP as the transport layer. After how much time will TCP start sending packets at the maximum sending rate? Illustrate your answer with a small graphic showing TCP's congestion window size in function of the number of elapsed RTT with the different TCP phases.

(c) (3pts) Explain why TCP's congestion control algorithm is labelled as *Additive-Increase-Multiplicative-Decrease* (AIMD).

Answers:

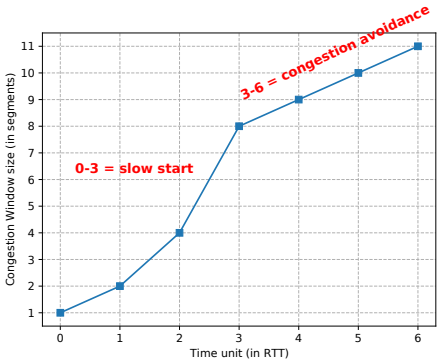
(a)

Here, the client will wait to receive an ACK before sending next packet:



Thus L bytes are sent for every L/R+RTT = 1100/1100 + 10 = 11 ms time period, that is a sending rate of 1100/11 = 100 bytes per ms = 100 KB/s.

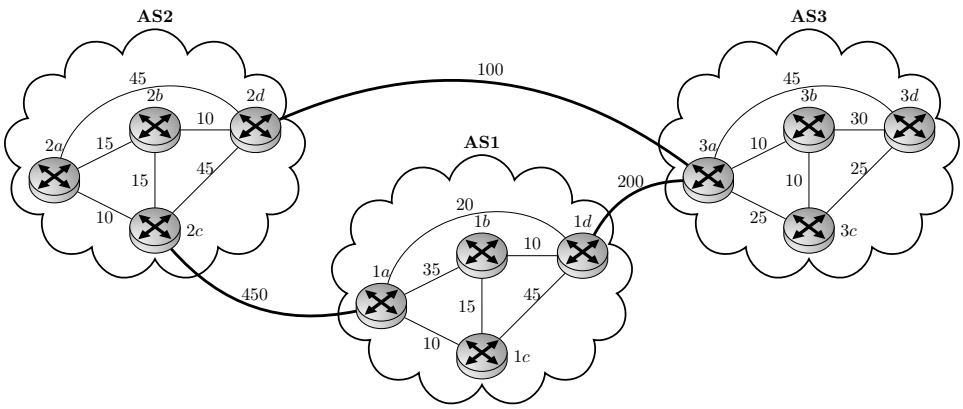
(b)



The answer is after 6 RTT. During the first timesteps, the window size doubles every RTT until reaching ssthresh, then it increases by 1 MSS every RTT. Once the window size reaches 11, the maximum sending speed has been reached as 11 packets will be sent in 1+10 = 11ms so a rate of 11*1100/11 bytes/ms = 1100 bytes/ms = 1.1 MB/s = R.

(c) After the slow start phase, TCP enters a Congestion Avoidance (CA) phase adding 1 MSS per RTT to the congestion window (hence "additive increase"). Under Reno's and more recent variants, TCP halves the congestion window size upon noticing a packet loss due to receiving 3 duplicate ACK segments (hence "multiplicative decrease"). After that, TCP will again enter a CA phase and will eventually reach another packet loss dividing the window size by 2, and so on and so forth. This gives TCP's congestion control its typical "saw-toothed behavior".

4. **Network Layer (10pts)**
Consider the following network topology:



The above network is made of 12 routers belonging to 3 Autonomous Systems (AS). The AS have been allocated with the following IP blocks **129.16.0.0/24** for AS1, **129.16.1.0/24** for AS2 and **129.16.2.0/24** for AS3. BGP is used for inter-AS routing while the 3 AS use an intra-AS routing protocol that minimizes the latency (displayed next to each link). No policy has been set for the route preferences by the administrators of the 3 AS, hence the rest of BGP's usual list of criteria is used here. We assume the routers use both destination-based forwarding and longest prefix matching.

(a) (3pts) Some links of the network will never get used by any traffic. List all such links.

(b) (3pts) Provide the routing table of router's 1c assuming all a-routers have one interface configured with an IP address with 1 has host part, b-routers with 2 as host part, c-routers with 3 and d-routers with 4 (to simplify, we assume 1 IP address only per router).

(c) (4pts) Let's suppose AS1 and AS2 decide to merge. What will be their new IP address block? Will this have consequences on the routers and the routing tables and which ones?

Answers:

(a) The links 2a-2d, 2c-2d, 3a-3d, 3a-3c, 1c-1d, 1a-1b will never be used as they can each be "replaced" by a path with a smaller cost. (Note that 2c-1a is used as BGP will favor the shortest AS-PATH between AS2 and AS1).

(b)

Destination Address (Dst IP)	Interface
129.16.0.3/32	1c (localhost)
129.16.0.1/32	1a
129.16.0.0/22	1b
129.16.1.0/24	1a

The row using 2b can also simply be marked as "default route".

(c) The merged block will be 129.16.0.0/23. The consequences on the routers is that 2c and 1a now become "internal routers" in BGP instead of border gateways, hence they do not need to keep an eBGP connection but an iBGP connection instead. Both AS1 and AS2 will have to use the same intra-AS protocol and all the routing tables will be recomputed. Since BGP favors shortest AS-PATH then use Hot Potato routing and the link 1a-2c has a very large latency, the individual tables should be identical afterwards except potentially at 3a which will only keep 1 route to go to AS1+AS2.

5. **Link Layer and Wireless (10pts)**

Let's consider a hypothetical link-layer protocol ALOHA2 that provides *error detection* and *multiple access* to a shared link. Error detection is obtained using single-bit parity check (even parity) by adding an extra parity bit for every seven bits of the original data. Multiple access is done using the slotted-ALOHA protocol using a probability of 1/2 for sending data during each timeslot. The goal of the exercise is to compare ALOHA2 with Ethernet and 802.11 link-layer protocols.

(a) (2pts) Review how a single parity-bit is working by giving 2 examples of 8-bits words: 1 byte where no error is detected and 1 byte that contains at least one error.

(b) (2pts) Now consider a short acknowledgment link-layer packet of **70 bytes** (excluding all error-detection bits), how many bytes will need to be sent with ALOHA2 and with Ethernet/Wifi (both using 32-CRC, a 32-bits Cyclic Redundancy Check)? What about sending a **1400 bytes** long packet?

(c) (2pts) Actually, **32-CRC** looks very similar to a message's hash. Why would it be though a very poor choice to sign messages?

(d) (4pts) ALOHA2 will not be very efficient to share the channel as the probability that one out of n senders successfully sends its transmission during one timeslot would be $n/2^n$. What are the mechanisms in place in Ethernet's protocol that allow it to be much more channel-efficient than ALOHA2? (No need to enter in details here, just mention the different mechanisms and compare them with ALOHA2). Are all of these mechanisms also used in Wifi's protocol and if not, why is that so?

Answers:

(a) The parity bit indicates the parity of the number of ones in the data part: 0 for even and 1 for odd. For example, receiving 00000000 will be labelled as "no error" as 0 ones = even = 0 for the last bit, whereas 10000000 contains an error somewhere as 1 is expected as the last bit (which itself can be wrong).

(b) For ALOHA2, 8*70/7 = 80 bytes and for Ethernet/Wifi 70+4 = 74 bytes only. For a 1400-packet, ALOHA2 will need to send 8*1400/7 = 8*200 = 1600 bytes whereas Ethernet/Wifi will send 1404 bytes.

(c) 32-CRC is used for random error detection and is not designed to be secure, it is for example easily reversible and collisions are very easy to produce. On top of that using 32-bits only is much too short to sign a message and secure hash-functions use at least 256+ bits.

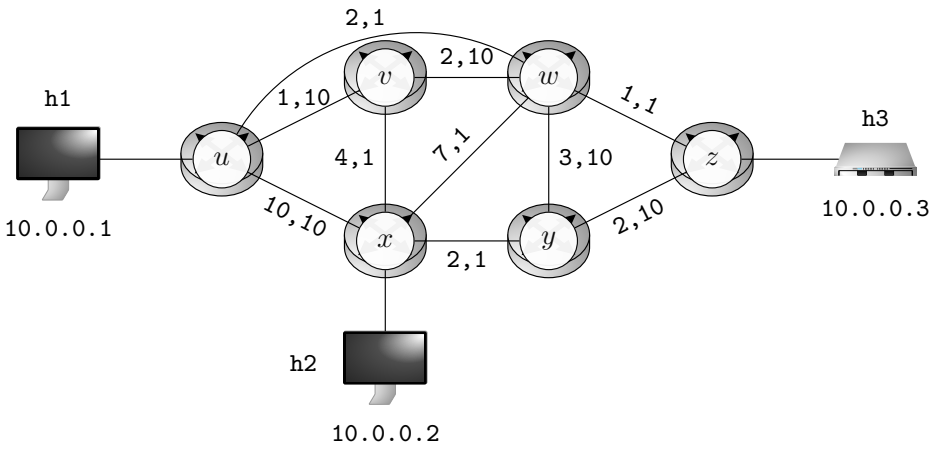
(d) Ethernet uses CSMA/CD that is based on:

- Carrier-Sense: Ethernet senses the channel before trying to send data (contrary to ALOHA2 only flipping a coin to decide to send or not).
- Collision Detection: Upon detected a collision, Ethernet will abort the transmission whereas ALOHA2 will always waste at least a full timeslot.
- Exponential Backoff: Ethernet will wait exponentially longer time before attempting to re-send while ALOHA2 will re-try every subsequent timeslot.

Only "Collision Detection" is not part of Wifi's protocol as it is difficult to both send and listen at the same time for wireless channels; 802.11 uses instead "Collision Avoidance".

6. **Software-Defined Networking (10pts)**

Let's consider the following network:



In the above network, each link connecting 2 routers is labelled ℓ, B where ℓ is a latency measure (in ms) and B is the bandwidth capacity of the link (in Gbps).

The 2 clients are both communicating with the server **h3** (with IP `10.0.0.3`) but each is rather interested in utilizing different routes: **h1** is interested by the route with lowest latency whereas **h2** would like to have the highest possible throughput for the connection. SDN will be used to make this possible and satisfy both clients.

(a) (4pts) Assume first that destination-based forwarding is in place on the routers. Draw a directed graph showing the paths used by packets with destination **h3** in the following two situations:

1. All the routes have been configured to use the lowest latency path (equivalent to running least-cost Dijkstra algorithm with latency ℓ as cost).
2. All the routes have been configured to use the maximum bandwidth path (equivalent in this graph to running least-cost Dijkstra algorithm with $1/B$ as cost).

That is, for each case, determine the next-hop router for packet with destination **h3**.

(b) (3pts) The administrator of the network decided to configure the routers so that, by default, the low-latency routes are used for routing packets based on their destination. On top of those already configured flows, we will add flows so that **h2** uses the high-bandwidth paths to reach **h3**. Copy and complete the following table with the necessary flows to add for this to happen:

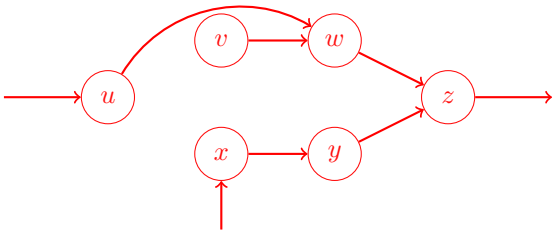
Switch to add a new flow	Match fields	Actions
...	...	...

Use eg IP,nw_src=... and IP,nw_dst=... to specify source/destination IP addresses and output:x to forward packet on the interface that is connected to x.

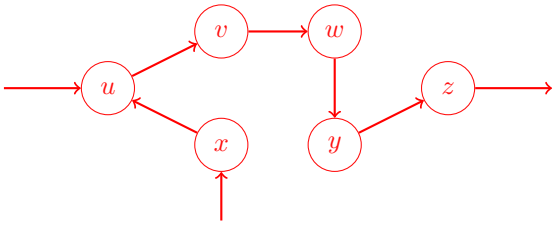
(c) (3pts) Suppose that **h1** was sending UDP traffic whereas **h2** was sending TCP traffic. Would it be possible to make the same paths that we configured using SDN but keeping destination-based forwarding and using some priority-based *Packet Scheduler* on each router? (here UDP packets can be one class of traffic and TCP another class).

Answers:

(a) For the low-latency paths:



For the high-bandwidth paths:



(b) 3 new flows must be added:

Switch	Match fields	Actions
x	IP,nw_src=10.0.0.2,nw_dst=10.0.0.3	output:u
u	IP,nw_src=10.0.0.2,nw_dst=10.0.0.3	output:v
w	IP,nw_src=10.0.0.2,nw_dst=10.0.0.3	output:y

(c) No! Packet scheduling may allow us to prioritize some traffic and for example here send UDP packets earlier than TCP packets when they are standing in one of the router's queue. However, without generalized-forwarding, only the destination will matter and since both traffic are going to the same destination (**h3**), they will use the same route. This cannot work here as for example at x, u and w, the routes differ!