



**Note:**

- During the attendance check a sticker containing a unique code will be put on this exam.
- This code contains a unique number that associates this exam with your registration number.
- This number is printed both next to the code and to the signature field in the attendance check list.

## Advanced Computer Networking

**Exam:** IN2097 / Retake

**Date:** Wednesday 3<sup>rd</sup> April, 2024

**Examiner:** Prof. Dr.-Ing. Georg Carle

**Time:** 08:00 – 09:15

### Working instructions

- This exam consists of **12 pages** with a total of **5 problems**.  
Please make sure now that you received a complete copy of the exam.
- The total amount of achievable credits in this exam is 75 credits.
- Detaching pages from the exam is prohibited.
- Allowed resources:
  - one **analog dictionary** English ↔ native language
- Subproblems marked by \* can be solved without results of previous subproblems.
- **Answers are only accepted if the solution approach is documented.** Give a reason for each answer unless explicitly stated otherwise in the respective subproblem.
- Do not write with red or green colors nor use pencils.
- Physically turn off all electronic devices, put them into your bag and close the bag.

Left room from \_\_\_\_\_ to \_\_\_\_\_ / Early submission at \_\_\_\_\_

## Problem 1 Quiz (17 credits)

The following questions cover multiple topics and can be solved independently of each other. The multiple choice questions need to be filled out as follows:

Mark correct answers with a cross



To undo a cross, completely fill out the answer option



To re-mark an option, use a human-readable marking



All multiple choice questions have exactly a single correct answer.

a)\* Which of the following is a correct IPv4 address in the common notation?

20.255.20.255

20.255.20.256

14.FF.14.FF

20:255:20:255

b)\* Which of the following statements about TCP CUBIC is correct?

It is more robust to packet loss than TCP BBR.

It tries to keep 1 BDP of data inflight.

It is a delay-based congestion control algorithm.

It keeps buffers on the path full.

c)\* Which of the following statements about addresses is correct?

The address space of Ethernet MAC addresses is **bigger** than the address space of **IPv6** addresses.

Ethernet uses a hierarchy for its MAC addresses.

IPv4 uses a hierarchy for its addresses.

The address space of Ethernet MAC addresses is **smaller** than the address space of **IPv4** addresses.

d)\* Which of the following statements about Software-Defined Networks (SDNs) is correct?

Not all network deployments using Network Function Virtualization (NFV) are SDNs.

Only software switches can be used in SDNs.

In an SDN context, all middleboxes are called router.

The P4 language is the successor of SDN.

e)\* Which of the following statements about DNS zones is **not** correct?

It starts and ends with a SOA record.

It has its own NS records.

It must have an A or AAAA records.

It must have a SOA record.

f)\* How many entries has TBL24 in the DIR-24-8 routing algorithm?

65 536

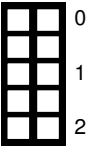
Size of TBL24 depends on the number of routing table entries.

16 777 216

4 294 967 296

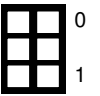
g)\* **What** does OUI stand for, **where** can it be found, and **what** does it represent?

- Organizationally Unique Identifier, first 3-bytes part of the MAC
- Unique vendor identifier to allow assignment of globally unique MAC addresses



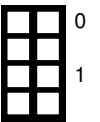
h)\* Name one approach from the lecture to detect QUIC deployments besides ZMap.

HTTP ALTSVC header or DNS HTTPS/SVCB records



i)\* NATs can artificially increase the IPv4 address space. Name and shortly explain one fundamental problem due to NAT discussed during the lecture.

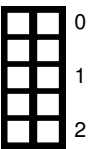
- No end-to-end associativity anymore
- Devices are not reachable from the Internet but have to initiate connections themselves.



j)\* Name and shortly explain two used IPv6 address types from the lecture.

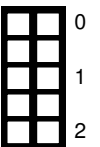
Two out of:

- Unicast: From one sender to exactly one receiver
- Multicast: From one sender to all (multiples or one) members in a group
- Anycast: From one sender to one member of a group



k)\* What limitation do CNAME records have and why is that relevant for the zone apex?

- It is a canonical name and therefore valid for all queried records
- If a CNAME exists no other record for the name can exist
- Zones start with a SOA record and hence CNAMEs are not available at the zone apex

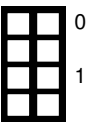


l)\* Argue what type of load balancing is indicated by the request response shown in Listing 1.

```
1 HTTP/1.1 301 Moved Permanently
  Location: http://www.google.de/
3 Content-Type: text/html; charset=UTF-8

5 <HTML><HEAD><meta http-equiv="content-type" content="text/html; charset=utf-8">
  <TITLE>301 Moved</TITLE></HEAD><BODY>
7 <H1>301 Moved</H1>
  The document has moved
9 <A HREF="http://www.google.de/">here</A>
  </BODY></HTML>
```

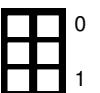
Listing 1: Response to a content request




HTTP-based load balancing: Redirect via HTTP status code.

m)\* Convert the IPv6 address 2001:0db8:0000:0000:8a2e:0000:0370:7334 to its shortened form.


2001:db8::8a2e:0:370:7334




## Problem 2 Traceroute and DNS (17 credits)

0 1  a)\* What problem does an authoritative name server have, when trying to perform load balancing based on the client's location without any DNS extension information?

- The authoritative name server only sees the resolvers IP address
- Depending on the resolver's granularity, the location of the resolver and the client differs.

0 1  b) Name and shortly explain the solution for the problem in Subproblem a) presented in the lecture.


- EDNS client subnet extension
- Resolver attaches the subnet of the client to the query
- Answers are only valid for the specified subnet and can only be used by the resolver for request from this subnet

0 1  c)\* In order to perform a traceroute to a domain name we need to resolve it to an IP address using DNS. In this problem, we will use IPv4 for our traceroute command. Which record type do we need to query for that?


- A records provide IPv4 address data

1	ns1.caida.org.	192.172.226.78
2	jungle.caida.org.	192.172.226.32
3	ns1.ucsd.edu.	128.54.16.2


Table 2.1: caida.org NS names and their corresponding IPv4 addresses

0 1  d)\* We plan to perform our traceroute towards *caida.org*. The zone of *caida.org* has three NS records. In Table 2.1, we list the names and the corresponding IP addresses. Why are the first two names (*ns1.caida.org* and *jungle.caida.org*) not enough according to the lecture?

- Both IP addresses are part of the same /24
- Therefore, topological diversity according to name server requirements are not given

0 1 2  e)\* Shortly explain the basic principle of traceroute and why it works.

- Sends out packets with increasing TTL starting at 1
- Each router decreases the TTL by 1
- When the TTL reaches 0, the router drops a packet and returns an ICMP time exceeded
- The src address of that packet is reported as the router address at the measured hop length

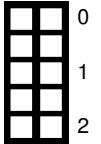
0 1  f)\* Name a reason why the traditional traceroute is not sufficient to always produce consistent results in modern day networks.

- Traditional traceroute does not consider load balancing
- Therefore, multiple measurements can uncover different paths

1	188.1.37.89	cr-gar1-be2-147.x-win.dfn.de	1ms	1ms	1ms
2	62.40.124.217	dfn.rt1.fra.de.geant.net	9ms	9ms	9ms
3	62.40.98.23	ae4.mx1.lon.uk.geant.net	21ms	21ms	21ms
4	163.253.1.118	core1.ashb.net.internet2.edu	165ms	163ms	164ms
5	163.253.1.114	core1.losa.net.internet2.edu	163ms	163ms	
	137.164.26.200	hpr-lax-agg10-i2.cenic.net	162ms		
6	137.164.26.43	hpr-100ge-sdg-hpr3.cenic.net	162ms	162ms	162ms
7	192.172.226.78	rommie.caida.org	165ms	165ms	165ms

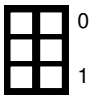
Table 2.2: Simplified traceroute starting from the TUM university network and targeted to caida.org.

g)\* Mark location hints in at least four of the domain names in Table 2.2. Alternatively, you can also reference the hop and write down the location hint in the solutionbox.



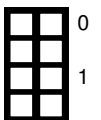
1 *gar1* The traceroute is starting from our Campus in Garching  
 2 *fra.de* Frankfurt, DE  
 3 *lon.uk* London, UK  
 4 *ashb* Ashburn, Virginia, US  
 5 *losa* Los Angeles, California, US  
 5 *lax* Los Angeles, California, US  
 6 *sdg* San Diego, California, US

h)\* Can you always trust the location hints in domain names? Briefly explain your answer!



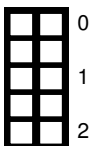
- IP addresses especially at borders are in some cases from the peering partner not the router operator itself
- The domain names are configured by the IP address owner which is in our case not the router operator.
- The domain names can also point to the other end of the link and not the interface's location itself

i)\* What is the reason for the high latency increase between Hops 3 and 4? Explain!



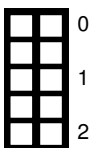
Transatlantic link- physical distance of more than 8700km- speed of light constraints

j)\* Explain the specialty visible on Hop 5 and why this poses a problem for the interpretation of traceroute results.



- Different addresses seen for the same hop length
- This is caused by load balancing
- Incorrect links might be inferred

k)\* Name and explain one reason, as covered in the tutorial, why the average latencies of Hops 5 and 6 are lower than the ones of Hop 4.



- ICMP throttling or lower ICMP prioritization under load
- Slower forwarding/responding when these methods are deployed
- Therefore, Hop 4 seems to have a lower priority than Hops 5 and 6 for this type of messages

### Problem 3 Reverse Hexdump (13.5 credits)

This problem investigates a captured Ethernet frame, shown in Figure 3.1. The frame was captured in between  $H_1$  and  $R$  as shown in Figure 3.2. For each interface, the MAC and IPv4 address is noted. Unfortunately, the frame was corrupted during the capture process. The lost bytes are marked with black areas in Figure 3.1.

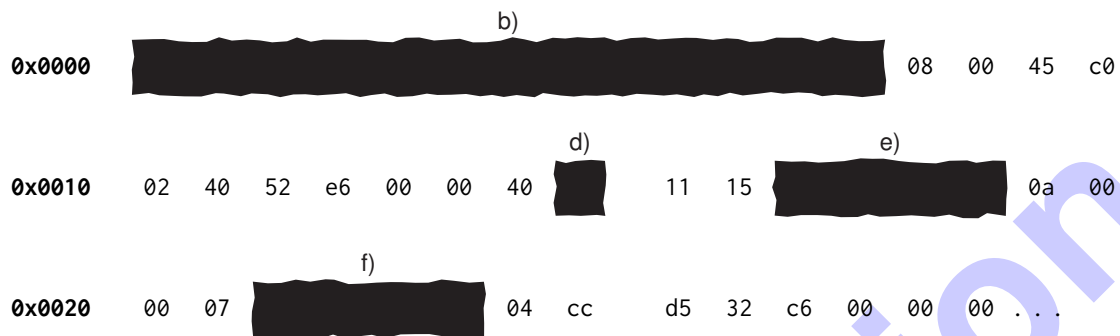


Figure 3.1: Hexdump of an Ethernet frame starting with the Ethernet header.

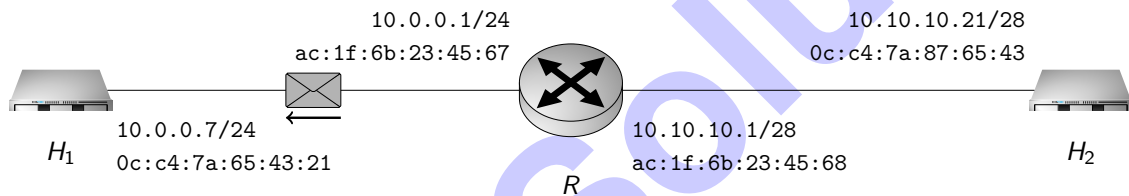


Figure 3.2: Network Topology

As you want to repair the captured hexdump, you have to reconstruct the corrupted parts. If a solution box contains a table, fill each cell with a single byte. You know that the captured frame belongs to an HTTP/3 over QUIC connection between  $H_1$  and  $H_2$ , where  $H_1$  is the client. You also know that the frame went in the direction indicated by an arrow.

- 0 


 1 

- a) In our case, only the captured frame was corrupted. If a bit-error occurred during the transmission, how could the receiver possibly detect it? Name a header field that is used for this purpose and explain how the value of this field is determined.

The receiver could possibly detect the error by using the Frame Check Sequence (FCS) field in the Ethernet header. The value of this field is determined by the sender by calculating a CRC checksum over the whole frame.

- 0 


 1 


 2 

- b) Reconstruct the first corrupted block of the captured frame, marked with an b).

Corrupted block: 

0c	c4	7a	65	43	21	ac	1f	6b	23	45	67
----	----	----	----	----	----	----	----	----	----	----	----

- 0 


 1 

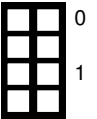
- c)\* Which **Layer 3** protocol is used in the captured frame? Please also mark and name the field in the hexdump that supports your answer.

Ethertype: [12, 13] = 0x0800 ⇒ IPv4

d) Reconstruct the second corrupted block marked with d) and explain why you chose the value.

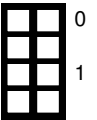
Corrupted block:

Reason: 0x11 is the protocol number for UDP.



e) Reconstruct the third corrupted block marked with e) and name the header field.

Corrupted block:     Header Field: Source IP Address

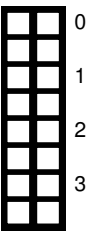


f) Which field(s) were present in the fourth corrupted block of the captured frame, marked with f)? For each field, state whether you are able to reconstruct the original content with the given information or not. If no, state the reason. If yes, provide the reconstructed content.

It contained the source port and destination port fields of the UDP header.

As we know that the IP packet contains a QUIC packet sent from server to client, the source port is 443 and therefore 01 bb.

The destination port (belonging to  $H_1$ ) is chosen randomly and therefore cannot be reconstructed.



g)\* Before  $H_1$  was able to send the first frame to  $R$ , it had to resolve the IP address of  $R$  to a MAC address. Which protocol was likely used for this purpose?

ARP was probably used, as IPv4 addresses are used.



h) In Subproblem b), you reconstructed the first 12 bytes of the Ethernet header. What would the reconstructed bytes look like in a frame sent from  $H_1$  to  $R$  to resolve the MAC address of  $R$  for the first time using the protocol from Subproblem g)?

First 12 bytes:



i)\* How many different IP addresses are contained in the subnet between  $H_2$  and  $R$ ?

Prefix is /28 =>  $2^{32-28} = 16$



### Problem 4 Network Calculus (14 credits)

This problem investigates delay bounds in networks using Network Calculus.

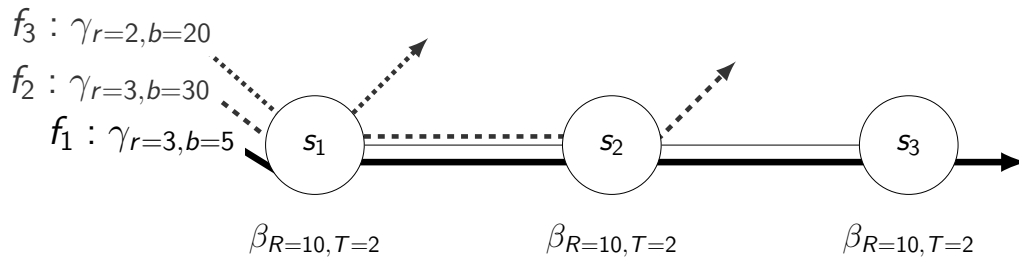


Figure 4.1: Topology with server- and flow specifications

Consider the topology in Figure 4.1. Assume each server employs strict priority queuing. Flow  $f_1$  has the lowest priority while Flows  $f_2$  and  $f_3$  have the highest priority. Flow  $f_1$  traverses three servers, Flow  $f_2$  traverses two, and Flow  $f_3$  traverses one.

We are interested in calculating the end-to-end delay bound of **Flow  $f_1$**  using the Separate Flow Analysis.

**Hint:**  $\beta^{l.o.} = \beta_{R-r, \frac{b+r \cdot T}{R-r}}$  and  $\alpha^* = \gamma_{r, b+r \cdot T}$

a)\* Perform the first step of the Separate Flow Analysis.

- Left-over service curve at  $s_1$ :  $\beta_{s_1}^{l.o.1} = [\beta_{10,2} - (\gamma_{2,20} + \gamma_{3,30})]^+ = [\beta_{10,2} - \gamma_{5,50}]^+ = \beta_{10-5, \frac{50+10 \cdot 2}{10-5}} = \beta_{5,14}$
- Output arrival curve of  $f_2$  after  $s_1$ :  $\alpha^* = \gamma_{3,30+3 \cdot 2} = \gamma_{2,36}$
- Left-over service curve at  $s_2$ :  $\beta_{s_2}^{l.o.1} = [\beta_{10,2} - \alpha^*]^+ = \beta_{10-3, \frac{36+10 \cdot 2}{10-3}} = \beta_{7,8}$
- Left-over service curve at  $s_3$ :  $\beta_{s_3}^{l.o.1} = \beta_{10,2}$

b) Perform the second step of the Separate Flow Analysis.

$$\beta_{e2e}^{l.o.} = \beta_{s_1}^{l.o.} \otimes \beta_{s_2}^{l.o.} \otimes \beta_{s_3}^{l.o.} = \beta_{\min(5,7,10), 14+8+2} = \beta_{5,24}$$

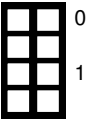
c) Perform the third step of the Separate Flow Analysis.

$$d_{e2e} = T_{e2e} + \frac{b}{R_{e2e}} = 24 + \frac{5}{5} = 25$$



d) Assume the following changes to the scenario in Figure 4.1:

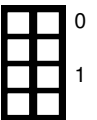
- The rate of Flow  $f_1$  is set to  $r = 4$
- The burst of Flow  $f_1$  is set to  $b = 500$



Compute the new delay bound of Flow  $f_1$  under the Separate Flow Analysis.

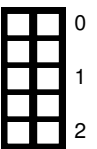
Check that left-over service curve has enough rate left to serve  $f_1$ :  $9 \leq 10$ .  
 Calculate delay bound:  $d = T + \frac{b}{R} = 24 + \frac{500}{5} = 124$

e)\* A flow with arrival curve  $\gamma_{r=5,b=10}$  is traversing a server with service curve  $\beta_{R=2,T=8}$ . Calculate the **delay bound** of the flow.



The delay bound is infinite because the rate of the flow is larger than the rate of the server

f)\* A flow with arrival curve  $\gamma_{r=5,b=480}$  is traversing 1 000 servers connected in series, each with the same service curve  $\beta_{R=20,T=2}$ . Calculate the **delay bound** of the flow. The method you choose should calculate a tight delay bound.



- Concatenate service curves:  $\beta_{e2e} = \beta_{\min(\bigcup_{s_i \in S} R_i), \sum_{s_i \in S} T_i} = \beta_{20, 1000 \cdot 2} = \beta_{20, 2000}$
- $d_{e2e} = T_{e2e} + \frac{b}{R_{e2e}} = 2000 + \frac{480}{20} = 2024$

g)\* Name one approach, besides deterministic network calculus, that is suitable for the analysis of systems with hard real-time criticalities.



Any one of: model checking, real-time calculus, trajectory approach, or schedulability analysis

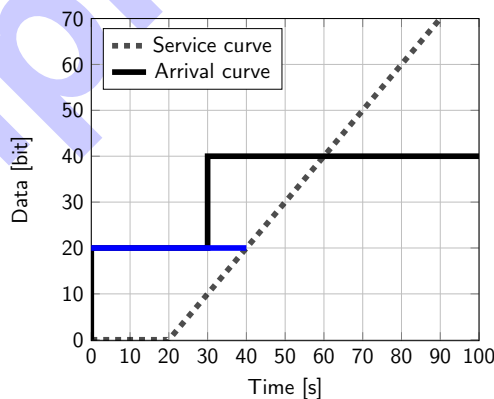


Figure 4.2: Arrival- and service curve

h)\* Consider the two curves in Figure 4.2. What is the delay bound of a flow with this **non**-token-bucket arrival curve traversing a server with this rate-latency service curve?



40

## Problem 5 Software-Defined Networking (13.5 credits)

This problem investigates a Software-Defined Network (SDN) powered by P4. For the following problems, consider the network given in Figure 5.1. Server 1 is configured to create and only accept tagged frames (VLAN ID 31), Server 2 only creates and accepts untagged frames. PCP and DEI are always set to 0. Switch A is a P4 switch handling the VLAN functionality. In the following subproblems, the P4 program running on Switch A is investigated. Listing 2 shows parts of the used P4 program.

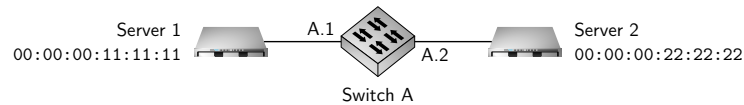
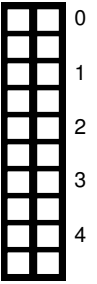


Figure 5.1: Network topology

```
2 header eth_t { bit<48> dstAddr;
3               bit<48> srcAddr;
4               bit<16> etherType; }
5
6 header veth_ext_t { bit<3> pcp;
7                   bit<1> dei;
8                   bit<12> vid;
9                   bit<16> etherType; }
10
11 struct standard_metadata_t { bit<16> ingress_spec;
12                             bit<16> egress_spec;
13                             // ...
14                             }
15
16 struct meta { /* unused */ }
17 struct headers { eth_t eth;
18                veth_ext_t veth_ext; }
19
20 parser ParserImpl(packet_in packet, out headers hdr, inout meta meta, inout standard_metadata_t
21 std_meta) {
22     // to be defined in Subproblem a)
23 }
24
25 control Pipeline(inout headers hdr, inout metadata meta, inout standard_metadata_t std_meta) {
26     action drop() {
27         mark_to_drop();
28     }
29
30     action untag(bit<16> egress) {
31         std_meta.egress_spec = egress;
32         hdr.eth.etherType = hdr.veth_ext.etherType;
33         hdr.veth_ext.[...]; // to be defined in Subproblem b)
34     }
35
36     action tag(bit<16> egress, bit<12> vid) {
37         std_meta.egress_spec = egress;
38         hdr.veth_ext.etherType = hdr.eth.etherType;
39         hdr.eth.etherType = 0x8100;
40         hdr.veth_ext.[...]; // to be defined in Subproblem c)
41         hdr.veth_ext.pcp = // to be defined in Subproblem c)
42         hdr.veth_ext.dei = // to be defined in Subproblem c)
43         hdr.veth_ext.vid = // to be defined in Subproblem c)
44     }
45
46     table forward {
47         actions = {
48             tag;
49             untag;
50             drop;
51         }
52         key = {
53             std_meta.ingress_spec: exact;
54         }
55         size = 4;
56         default_action = drop;
57     }
58
59     apply {
60         if (hdr.eth.isValid()) {
61             forward.apply();
62         }
63     }
64 }
65
66 // ...
```

Listing 2: VLAN P4 program

a)\* Write a parser using the P4 language for the P4 program in Listing 2. The parser starts at the start state and must be able to accept tagged and untagged Ethernet frames.



```

parser ParserImpl(...) {
  state start {
    transition parse_ethernet;
  }
  state parse_ethernet {
    packet.extract(hdr.eth);
    transition select(hdr.eth.etherType) {
      0x8100: parse_eth_ext;
      default: accept;
    }
  }
  state parse_eth_ext {
    packet.extract(hdr.veth_ext);
    transition accept;
  }
}

```

b)\* Complete the untag() action of Listing 2 (Line 27).

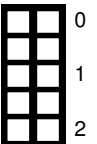


```

hdr.veth_ext.setInvalid();

```

c)\* Complete the tag() action of Listing 2 (Lines 33–36).

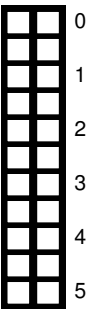


```

hdr.veth_ext.setValid();
hdr.veth_ext.pcp = 0;
hdr.veth_ext.dei = 0;
hdr.veth_ext.vid = vid;

```

d)\* The P4 program cannot work correctly without table data containing correct forwarding rules. Give the rules for Switch A, to correctly encapsulate and forward frames of Servers 1 and 2. Use the information given in Figure 5.1.



Match field(s)	Key	Action	Action data
std_meta.ingress_spec	A.2	tag	egress=A.1 vid=31
std_meta.ingress_spec	A.1	untag	egress=A.2



e)\* Headers in P4 can be validated and invalidated. What is the effect of the (in-)validation on the deparser.

The deparser reassembles packets only out of the valid headers. Headers that are invalidated are not considered by the deparser.

**Additional space for solutions—clearly mark the (sub)problem your answers are related to and strike out invalid solutions.**