Chair of Network Architectures and Services
Department of Informatics
Technical University of Munich

Esolution
Place student sticker here

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:**	Tuesday 16th April, 2019
Examiner:	Prof. Dr.-Ing. Georg Carle	**Time:**	15:30 – 16:30

	P 1	P 2	P 3	P 4	P 5
I					
II					

Working instructions

- This exam consists of
  - **16 pages** with a total of **5 problems** and
  - a two-sided printed **cheat sheet**.

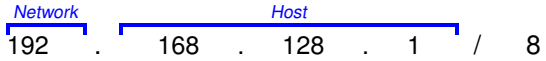
  Please make sure now that you received a complete copy of the exam.
- Detaching pages from the exam is prohibited.
- Subproblems marked by * can be solved without results of previous subproblems.
- **Answers are only accepted if the solution approach is documented.** Give a reason for each answer unless explicitly stated otherwise in the respective subproblem.
- Do not write with red or green colors nor use pencils.
- The total amount of achievable credits in this exam is 60 credits.
- Allowed resources:
  - one **analog dictionary** English ↔ native language
- 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 (9 credits)

0
1

a)* Clearly mark the network part and the host part of the given subnet address.



0
1

b)* Briefly explain the concept of 6to4.

Encapsulation of IPv6 packets in IPv4 packets to allow IPv6 communication via IPv4-only hosts.

0
1

c)* Given the routing table in Table 1.1, over which interface is a packet with destination IP address 10.10.10.10 forwarded under the assumption of LPM?

Prefix	Next Hop	Interface
192.168.10.0/30	192.168.10.2	eth0
10.10.10.0/28	192.168.10.3	eth1
10.10.10.0/24	10.10.10.2	wlan0
0.0.0.0/0	10.10.10.4	eth2

Table 1.1: Routing table

eth1
The destination IP address matches to the last three entries. According to LPM the second entry is selected.

0
1

d)* Name one advantage of using consistent hashing when distributing multiple clients to multiple content servers.

Less clients need to be remapped on average when the number of servers changes.

0
1

e)* Briefly explain the basic mechanism and goal of Certificate Transparency.

Publicly log all issued certificates to detect mis-issuance.

0
1

f)* Explain the main reason why the spanning tree protocol is used in switched networks.

Avoid forwarding loops.

g)* In IPv4 the 5-tuple consisting of addresses, ports, and protocol type is used to identify flows. What is used for IPv6?

0 1

In IPv6 the combination of the source address and the flow label from the header is used.

h)* TCP BBR enters Probe RTT about each ten seconds. Briefly explain what happens during this phase?

0 1

BBR reduces its inflight data to a minimum (4 packets) to drain all queues on the path.
This allows it to measure a low RTT sample.

i)* Assume Network Calculus uses a token bucket as arrival curve α and a rate-latency function as service curve β. Clearly mark the burstiness and processing delay parameters directly in Figure 1.1.

0 1

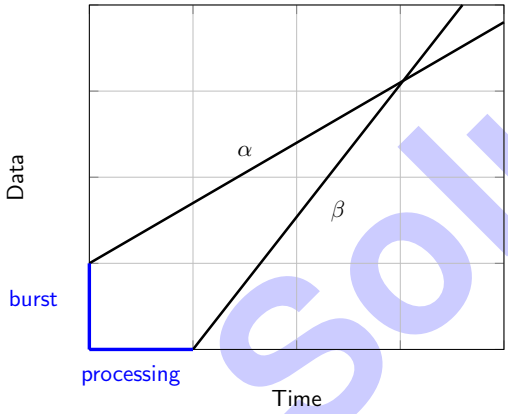


Figure 1.1: Arrival and service curve at a network node.

Problem 2 AS - BGP (12.5 credits)

This problem investigates AS level routing and BGP. Figure 2.1 shows an AS topology. The directed edges depict customer → provider relations. The dashed edges (- - -) depict peering relations.
Every AS wants to get connectivity for its owned prefixes and wants to earn/save money by forwarding traffic to their customers/peers.

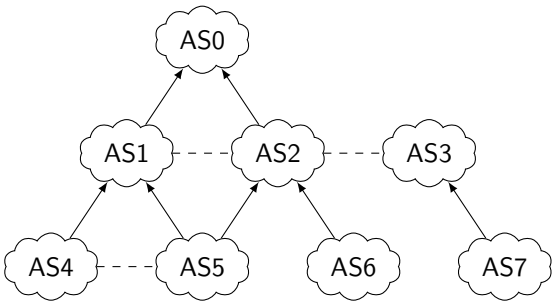


Figure 2.1: AS topology

0 1

a) Explain hot potato routing and name one non-monetary consequence.

Hand over traffic destined for another AS as soon as possible to minimize costs. Leads to asymmetric routing.

0 1

b) List all nodes that are part of the core of the topology in Figure 2.1 as calculated by the k-core algorithm. Assume all edges are bi-directional.

AS0, AS1, AS2, AS5, AS4

0 1

c) Table 2.1 shows the sources and destinations of traffic. Complete the table by providing the path the traffic is most likely routed along, assuming normal BGP behavior.

Source → Destination	**Path**
AS6 → AS4	AS2, AS1, AS4
AS7 → AS0	AS7 receives no announcement for the prefix of AS0
AS7 → AS6	AS3, AS2, AS6

Table 2.1: Paths taken by traffic

0 1

d) Explain whether or not AS5 announces the prefixes of AS4 to AS2, after learning them from AS4.

AS5 does not announce prefixes of AS4 since AS5 would need to pay AS2 for traffic destined to AS4 while not receiving any compensation (peering).

e) AS5 receives the announcements in Table 2.2. Decide whether or not each announcement gets accepted **independent of any previous announcements**. Justify your choice.

0
1

Announced path	Accept	Reason
AS1, AS0	Yes	Normal upstream sequence
AS2, AS0, AS1, AS2	No Yes	Routing loop Routing loop, but AS5 not in path
AS1, AS2, AS0	Yes	Upstream sequence with peering

Table 2.2: BGP announcements seen at AS5

f) Assume an AS with ASN 1111 receives an announcement with the following path:

AS2222, AS3333, AS4444, AS4444, AS5555, AS6666

Argue whether or not this is a valid AS path and explain the underlying concept and why it is used.

0
1

It is accepted because it is a normal sequence with path prepending. An AS uses path prepending when it concatenates its own ASN multiple times to the AS path in order to make it less attractive for other ASes to choose this path.

g) A regular BGP update contains four default fields. Name all four fields.

0
1
2

- Prefix
- AS Path
- Next Hop
- Origin

h) An attacker controlling AS5 wants to sniff traffic exchanged between AS1 and AS4. The attacker plans to get access to the traffic by routing it over AS5. Explain how and why the attacker could perform such an attack. Consider both traffic directions.

0
1
2
3

- AS1 $\rightarrow$ AS4: AS5 announces multiple more specific prefixes of AS4's prefix. This route is chosen because of LPM.
- AS4 $\rightarrow$ AS1: AS5 announces the prefix of AS1. This route is chosen by AS4 because it is less expensive (peering).

Problem 3 Reverse Hexdump (12 credits)

For this problem consider the network topology given in Figure 3.1. The topology consists of two hosts H_1 and H_2 which are interconnected by the switch S. The MAC addresses as well as the IPv4 addresses for each interface are given.

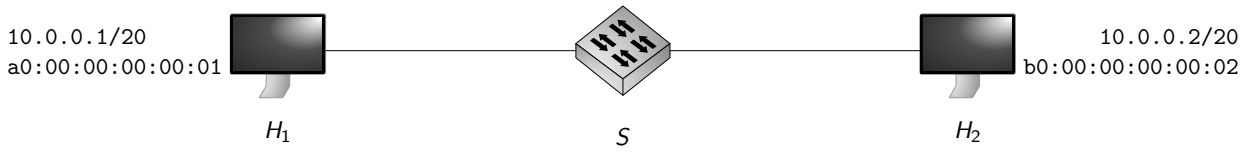


Figure 3.1: Topology

Now H_1 wants to send a message to H_2 using the Ping tool. Since no messages have been exchanged so far H_1 does not know the MAC address of H_2. To resolve this issue H_1 sends out an ARP Request. In the following you have to fill the packet headers using the given information. If no information is given for a field, make a reasonable choice.

`0x 01 0c ab ff` ⇒

Example	01	0c	ab	ff		

Each square can be filled with one byte in hexadecimal notation. For each header you have an additional preprint. Make clear which preprint contains your solution.

0 1

a)* Fill the fields of the Ethernet header.

Dst MAC	ff	ff	ff	ff	ff	ff
Src MAC	a0	00	00	00	00	01
Ethertype	08	06				

Dst MAC	ff	ff	ff	ff	ff	ff
Src MAC	a0	00	00	00	00	01
Ethertype	08	06				

0 1 2 3

b)* Fill the given fields of the ARP header.

Hardware Address Length	06					
Protocol Address Length	04					
Sender Hardware Address	a0	00	00	00	00	01
Sender Protocol Address	0a	00	00	01		
Target Hardware Address	00	00	00	00	00	00
Target Protocol Address	0a	00	00	02		

Hardware Address Length	06					
Protocol Address Length	04					
Sender Hardware Address	a0	00	00	00	00	01
Sender Protocol Address	0a	00	00	01		
Target Hardware Address	00	00	00	00	00	00
Target Protocol Address	0a	00	00	02		

Space for computations.

0 1 2

c)* Now that H_1 knows the hardware address of the next hop it can send out the Ping message. Fill the given fields of the IPv4 header of the Ping.

TTL	40					
Protocol	01					
Src Address	0a	00	00	01		
Dst Address	0a	00	00	02		

TTL	40					
Protocol	01					
Src Address	0a	00	00	01		
Dst Address	0a	00	00	02		

d)* Who will reply to the ARP request and the Ping message?

0 1

ARP Request: H_2 Ping: H_2

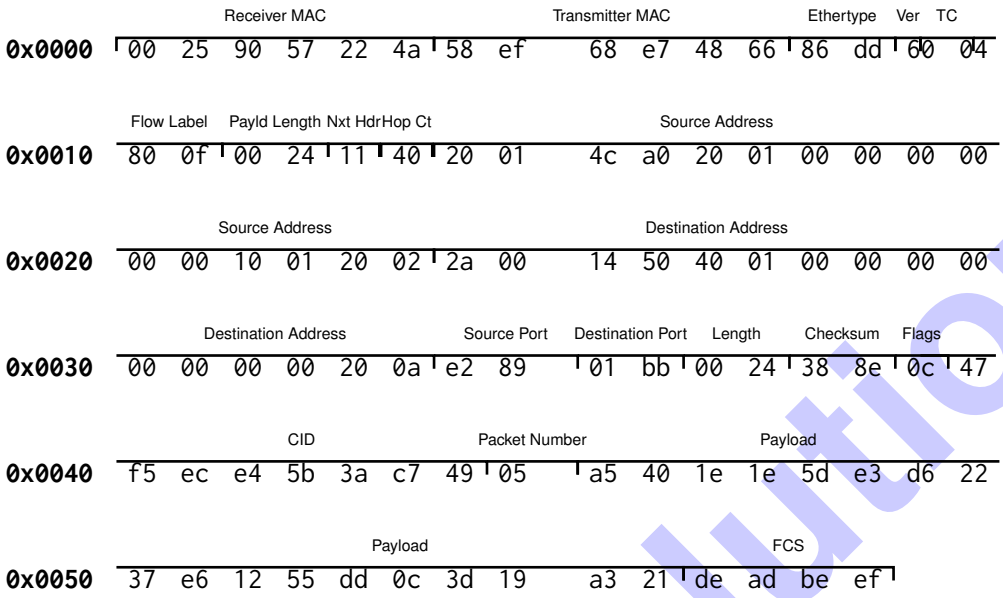


Figure 3.2: Hexdump

For the following consider the hexdump in Figure 3.2. It shows a fully captured Ethernet frame.

e)* Name all protocols contained in the hexdump.

0 1 2

Ethernet, IPv6 (Ethertype `0x86dd`),
UDP (Next Header `0x11`), QUIC (UDP port `0x01bb` = 443)

f)* The actual payload is marked in the hexdump. Compute the share of the payload to the overall transmitted number of bytes.

0 1

Payload size: 18 Bytes
Frame size: 94 Bytes

$\frac{18}{94} = \frac{9}{47} \approx 19.1\%$

g)* The FCS is used to detect bit errors during the transmission. Briefly explain why it cannot be used to prevent anyone from modifying arbitrary fields in the frame.

0 1

The FCS is computed using CRC. Anyone who can modify the frame can also compute the new checksum and append it to the modified frame.

Problem 4 Software-Defined Networking (12 credits)

This problem investigates a Software-Defined Network (SDN) powered by OpenFlow.

0 1 2

a)* An OpenFlow switch can forward a packet either based on its local ruleset or by requesting a decision from the controller. Explain two advantages of using the local ruleset.

- Using the local ruleset leads to a faster forwarding decision than contacting a potentially remote controller
- Using the local ruleset lowers the load on the controller

0 1 2

b)* An SDN uses different planes (management, control, and data plane) for different tasks. Explain to which plane OpenFlow and P4 belong respectively.

- P4 operates on the data plane as it is a programming language specifying the behavior of the data plane.
- OpenFlow defines a protocol between control and data plane, therefore it does not belong to a single plane but connects data and control plane.

For the following problems consider the network given in Figure 4.1. Switch S is an OpenFlow-enabled switch, attached to a controller. Switch S is configured to drop any packet which does not match against any of the currently installed rules (see Listing 1). The MAC addresses of all network nodes are statically configured and correct, i. e. no protocols for address resolution are required. The IP addresses of all network nodes are statically configured.

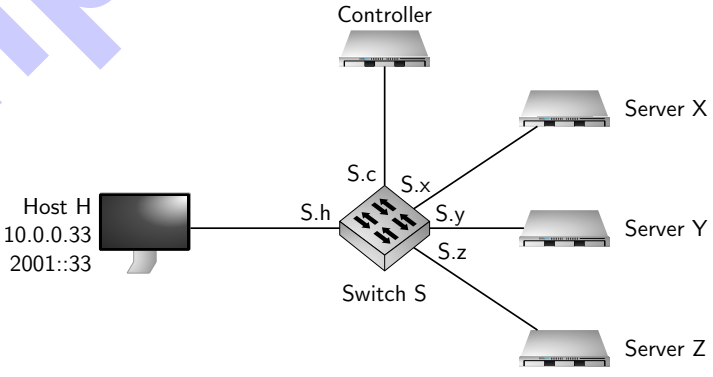


Figure 4.1: Network topology

```
1 ovs-ofctl add-flow S dl_type=0x0800,nw_dst=10.0.0.1,nw_proto=0x6,tp_dst=80,actions=output:S.y
  ovs-ofctl add-flow S dl_type=0x86dd,nw_dst=2001::1,nw_proto=0x6,tp_dst=443,actions=output:S.z
3 ovs-ofctl add-flow S dl_type=0x86dd,nw_dst=2001::42,nw_proto=0x11,tp_dst=443,actions=output:S.x
```

Listing 1: OpenFlow rules installed on S

Remark: `nw_proto` specifies the protocol transported as payload of the network layer (see IP protocol numbers on the cheatsheet), `tp_dst`/`tp_src` specifies the destination/source port on the transport layer.

c)* Enter the correct values for Table 4.1 using the ruleset given in Listing 1. The table should contain the name of the server, its IP-address, the name of the used transport protocol, the port addressed by the transport protocol, and a sensible choice for the application layer protocol.

0 1 2 3

Server	IP address	Transport protocol	Port	Application layer protocol
X	2001::42	UDP/(QUIC)	443	HTTPS
Y	10.0.0.1	TCP	80	HTTP
Z	2001::1	TCP	443	HTTPS

Table 4.1: Information about servers and protocols in use

d) Host H wants to use the application on Server Y, but cannot establish a connection, despite using the correct addresses and ports. Briefly explain why, using the rules of Listing 1.

0 1 2

- Packets sent from Host H to Server Y arrive at the server (rule 1 applies)
- Packets in reverse direction do not have matching rules and are discarded by Switch S, therefore no connection can be established.

e) Specify an OpenFlow rule fixing the problem described in Problem d). The rule should be as restrictive as possible, i.e. only Host H should be allowed to successfully connect to Server Y and only the specified application should be allowed to communicate.

0 1 2 3

```
ovs-ofctl add-flow S dl_type=0x0800,nw_dst=10.0.0.33,nw_proto=0x6,tp_src=80,actions=output:S.h
```

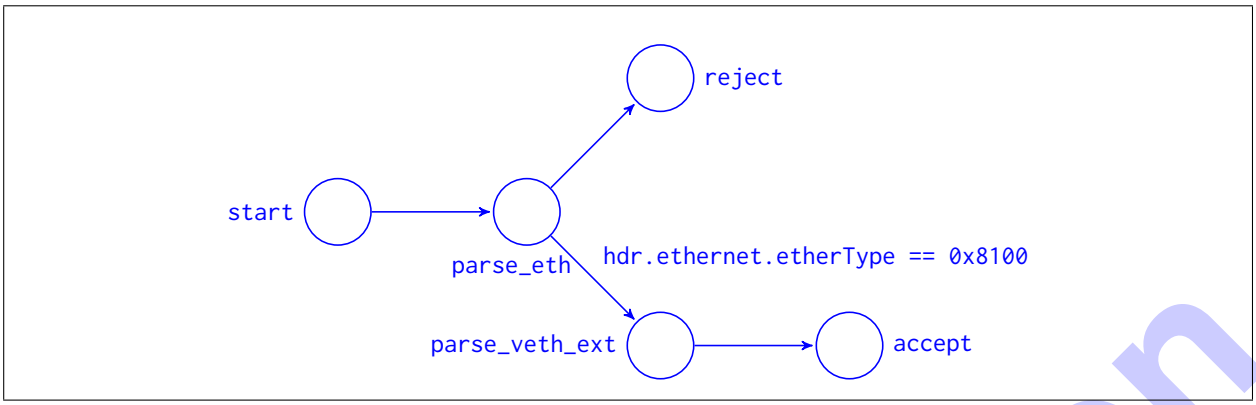
Problem 5 P4 Switching (14.5 credits)

This problem investigates a Software-Defined Network (SDN) powered by P4. The source code of a P4 switch program is given in Listing 2.

```
header eth_t       { bit<48> dstAddr;
                     bit<48> srcAddr;
                     bit<16> etherType; }
header veth_ext_t  { bit<3> pcp;
                     bit<1> dei;
                     bit<12> vid;
                     bit<16> etherType; }
struct std_meta    { bit<16> ingress_port; }
struct meta        { //unused
                   }
struct headers     { eth_t eth;
                     veth_ext_t veth_ext; }

parser ParserImpl(packet_in packet, out headers hdr, inout meta meta, inout standard_metadata_t
    std_meta) {
  state parse_eth {
    packet.extract(hdr.eth);
    transition select(hdr.ethernet.etherType) {
      16w0x8100: parse_veth_ext;
      default: reject;
    }
  }
  state parse_veth_ext {
    packet.extract(hdr.veth_ext);
    transition accept;
  }
  state start {
    transition parse_eth;
  }
}

control Pipeline(inout headers hdr, inout metadata meta, inout standard_metadata_t std_meta) {
  action drop() {
    mark_to_drop();
  }
  action decap(bit<16> egress) {
    std_meta.egress_port = egress;
    hdr.eth.etherType = hdr.veth_ext.etherType;
    hdr.veth_ext.setInvalid();
  }
  table forward {
    actions = {
      decap;
      drop;
    }
    key = {
      std_meta.ingress_port: exact;
      hdr.eth.srcAddr: exact;
      hdr.veth_ext.vid: exact;
    }
    size = 4;
  }
  apply {
    if (hdr.veth_ext.isValid()) {
      forward.apply();
    }
  }
}

control DeparserImpl(packet_out packet, in headers hdr) {
  // see Problem c)
}
```

Listing 2: VLAN P4 program

a)* Visualize the parse graph of Listing 2 as state machine. Annotate the nodes with the according names and the non-trivial edges with the matches performed for this state transition.



For the following problems consider the network given in Figure 5.1. Switches 1 and 2 are VLAN capable switches. Servers A and C share a common VLAN (ID 16), Servers B and D also share a common VLAN (ID 32). All servers use regular Ethernet frames without any VLAN information. Ethernet frames with VLANs are only exchanged between the two Switches 1 and 2.

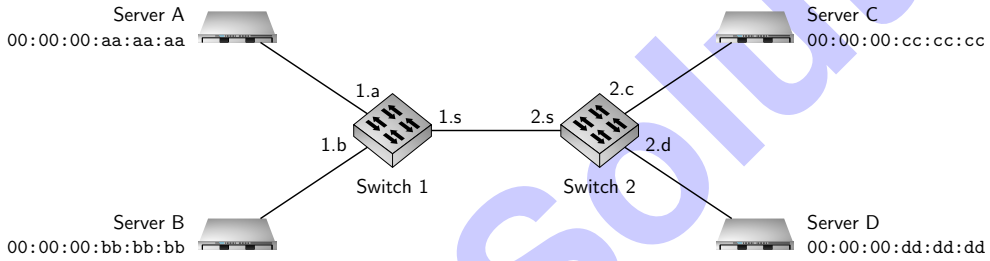


Figure 5.1: Network topology

b)* The P4 program cannot work correctly without table data containing correct forwarding rules. Give the rules for Switch 1, to correctly decapsulate frames received over Switch 2 from Servers C and D. Frames not originating from Servers C and D should be dropped. Use the information given in Figure 5.1. Use $*$ to mark the table cells which match on any value for which no more specific entry exists in this table.

Match field(s)	Key	Action	Action data
std_meta.ingress_port hdr.eth.src hdr.veth_ext	1.s 00:00:00:cc:cc:cc 16	decap	egress=1.a
std_meta.ingress_port hdr.eth.src hdr.veth_ext	1.s 00:00:00:dd:dd:dd 32	decap	egress=1.b
default	*	drop	*

0
1

c)* Create a valid deparser for the P4 program in Listing 2.

```
control DeparserImpl(packet_out packet, in headers hdr) {
  apply { packet.emit(hdr.eth);
          packet.emit(hdr.veth_ext);}
}
```

0
1
2
3

d)* Fill out the truth table for different P4 match expressions.

Match type	Match value	1111_2	1101_2	1011_2	1010_2
`exact`	`0xC`	false	false	false	false
`ternary`	`*01*`	false	false	true	true
`lpm`	`0xC/3`	false	true	false	false

0
1
2

e)* The decap action performs: `hdr.eth.etherType = hdr.veth_ext.etherType;`. Explain why this is necessary.

`hdr.eth.etherType` is originally set to `0x8100` meaning that VLAN is used. For untagging the frame the etherType of the VLAN payload is written back to the original VLAN frame before removing the VLAN header fields.

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