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 / Endterm	**Date:**	Friday 1st March, 2019
Examiner:	Prof. Dr.-Ing. Georg Carle	**Time:**	10:30 – 11:30

	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 (8 credits)

0
1

a)* Consider HTTP load balancers. Explain the interaction between load balancer and client.

HTTP load balancers redirect clients to a content server

0
1

b)* Figure 1.1 shows the arrival curve α and the service curve β at a network node. Clearly mark the delay bound and the backlog bound directly in Figure1.1.

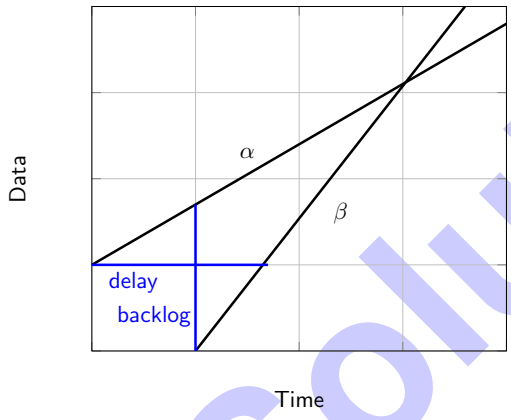

Figure 1.1: Arrival and service curve at a network node.

0
1

c)* Name the protocol used by Internet Protocol Security (IPsec) to establish Security Associations (SA) during the handshake.

IKEv{1, 2}

0
1

d)* Explain the concept of path prepending in BGP.

- AS includes own ASN multiple times in AS path
- Longer AS path makes route less attractive

0
1

e)* Explain the difference between the transmission (or serialization) delay and the propagation delay of a frame.

- Transmission delay: Time for node to send bits into link.
- Propagation delay: Delay introduced by physical link between nodes.

f)* Figure 1.2 shows an AS network. The arrows represent Customer→Provider relations, the dashed lines Peering agreements. Which ASes are Stub- and/or Multi-Homed AS? Fill the given table with the correct AS numbers.

0 1

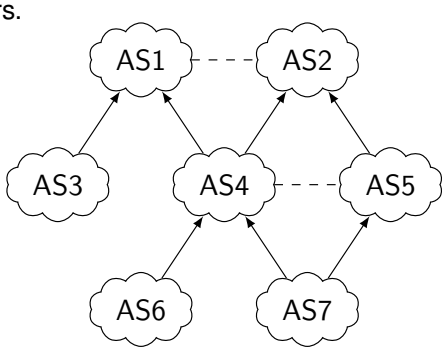

Figure 1.2: AS Topology

Tier-1 provider:	AS1, AS2
Stub-AS:	AS3, AS6
Multi-Homed AS:	AS7

g)* You try to build up a QUIC connection to `google.com` using your favorite browser, the *Internet Explorer*. Does this work? Give a brief explanation.

0 1

No, since QUIC hast to be supported by the client (browser).
(This is so far only possible with Chrome/ium and Opera.)

h) Table 1.1 shows two IPFIX flows. Assume the IP 5-tuple is used as flow identifier. Briefly explain whether or not the two flows can be merged using bidirectional flow matching with an unlimited active timeout and an inactive timeout of 30 seconds.

0 1

Start	**End**	**Src IP**	**Dst IP**	**Protocol**	**Src Port**	**Dst Port**	**Bytes**	**Packets**
t+0s	*t*+20s	10.10.10.10	8.8.8.8	TCP	2222	443	750	3
t+5s	*t*+32s	8.8.8.8	10.10.10.10	TCP	2222	443	4500	3

Table 1.1: Two IPFIX flows

The flows can not be merged because the source and destination ports are the same while the source and destination IP addresses are different.

Problem 2 Wireshark (11 credits)

According to the OSI model network protocols are distributed to seven different layers each containing several protocols. In this problem a packet is analyzed referring to the involved protocols.

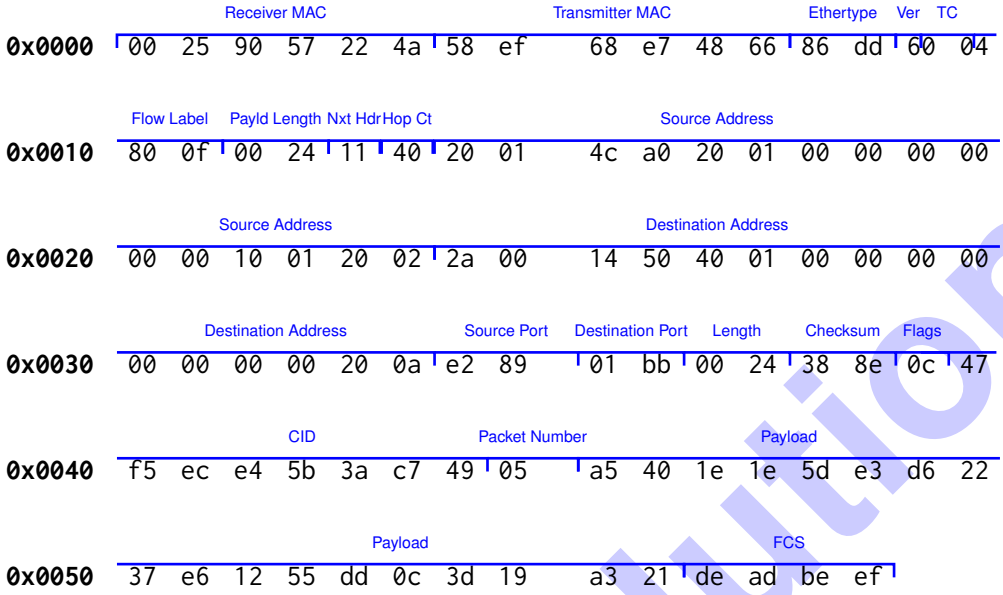

Figure 2.1: Hexdump of a complete Ethernet frame including FCS

a)* Mark the FCS in the hexdump. What is the purpose of the FCS?

Detect bit errors

In the next three subproblems you are asked to identify which protocols were used for each layer. For each question do the following:

- mark the corresponding bytes in the hexdump
- write the corresponding bytes in the solutionbox
- name the used protocol.

b)* Identify the L3 protocol.

Ethertype: `0x86dd`
IPv6

c) Identify the L4 protocol.

Next Header: `0x11`
UDP

d) Name the L4 destination port in decimal notation. Derive the type of the L4 payload from it.

`0x01bb`= Port 443
QUIC

e) Mark and name all fields of the L4 payload in Figure 2.1.

0 1 2

f)* For most flows in the Internet (also the given hexdump) inspecting the payload above L4 does not provide further insights. Briefly explain why.

0 1

Modern Internet protocols usually encrypt their payload. For example, HTTPS and QUIC use TLS.

g)* Name all parts of the IPv4 5-tuple, which is used to identify flows.

0 1 2

Source IP, Destination IP, L4 Protocol, L4 Source Port, L4 Destination Port

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

Problem 3 TCP (14 credits)

The Transmission Control Protocol is used by many different network applications. In this problem properties of the protocol are analyzed.

0 1

a)* Name two ways how TCP can detect lost segments?

- Duplicate ACKs
- Retransmission timeout

0 1

b)* The following sequence numbers arrive at an endpoint of a TCP connection. For each segment the receiver replies with an acknowledgment. Complete the missing ACK numbers. You can assume equal sized segments, a sufficiently large receiver buffer, and no influences due to flow control.

SEQ	5	6	8	9	7	10
ACK	6	7	7	7	10	11

0 1

c)* Figure 3.1 shows how the delivery rate of a TCP connection changes with increasing data inflight in an ideal environment. Update Figure 3.2 accordingly.

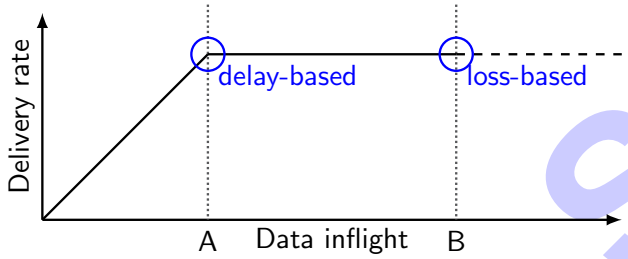

Figure 3.1

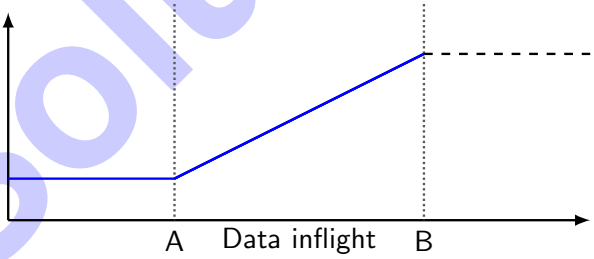

Figure 3.2

0 1

d)* How large is the amount of inflight data at the labels A and B in Figure 3.1?

A: BDP

B: BDP + bottleneck buffer size

0 1

e)* Mark and label the operation point of loss-based and delay-based congestion control algorithms in Figure 3.1.

f)* Briefly explain which amount of inflight data is optimal for TCP.

0 1

The optimal operation point is with one BDP of data inflight. This allows maximal delivery rate with minimal RTT.

0 1 2

g)* Briefly explain what happens during TCP BBR's Probe BW and Probe RTT phases.

- Probe BW: BBR periodically increases its sending rate to probe if more bandwidth is available
- Probe RTT: BBR reduces its inflight data to drain all queues and measure a correct RTT sample

The following questions relate to the topology shown in Figure 3.3. It shows two servers S_1 and S_2 and one host H. The bandwidth and round trip propagation delay are depicted next to each link.

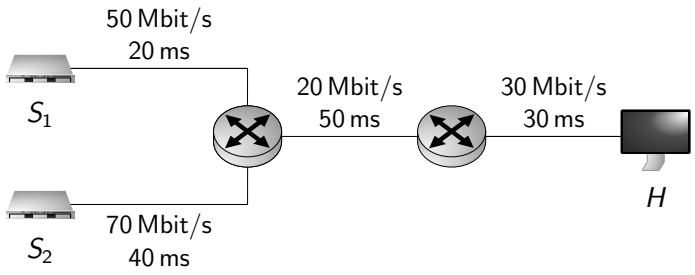

Figure 3.3: Topology

h)* Compute the bandwidth-delay product of the path between S_1 and H in kbit.

0 1 2

$$RTprop = \sum_{link \in path} RTprop_{link} = 20\,\text{ms} + 50\,\text{ms} + 30\,\text{ms} = 100\,\text{ms} = 0.1\,\text{s}$$

$$BtlBw = min_{link \in path} BtlBw_{link} = 20\,\text{Mbit/s}$$

$$BDP = RTprop \cdot BtlBw = 0.1\,\text{s} \cdot 20\,\text{Mbit/s} = 2\,\text{Mbit} = 2000\,\text{kbit}$$

Now two TCP flows F_1 (from S_1 to H) and F_2 (from S_2 to H) are started at the same time. Each flow transmits a file of size 125 MB (1 MB = 1×10^6 B). F_1 finishes the transmission after 100 s, F_2 after 200 s.

i)* Compute the average transmission rate for F_1 and F_2 in Mbit/s.

0 1

$$F_1: \frac{125\,\text{MB}}{100\,\text{s}} = \frac{1000\,\text{Mbit}}{100\,\text{s}} = 10\,\text{Mbit/s}$$

$$F_2: \frac{125\,\text{MB}}{200\,\text{s}} = \frac{1000\,\text{Mbit}}{200\,\text{s}} = 5\,\text{Mbit/s}$$

Jain's Fairness Index can be computed using the following formula.

$$\mathcal{F} = \frac{\left(\sum_i x_i\right)^2}{n \cdot \sum_i x_i^2} \quad \text{with } x_i \text{ being the bandwidth of flow } i \text{ and } n \text{ the number of flows}$$

j)* Name two properties of Jain's index.

0 1

scale free, arbitrary number of flows, normalized (between 0 and 1), $\frac{k}{n}$ if k flows are fair and $n - k$ receive nothing

k) Compute Jain's Index for F_1 and F_2.

0 1

$$\mathcal{F} = \frac{(10+5)^2}{2 \cdot (10^2 + 5^2)} = \frac{225}{250} = \frac{9}{10} = 0.9$$

Problem 4 Traceroute (13.5 credits)

This problem investigates the traceroute tool.

0
1
2
3

a)* Figure 4.1 shows a network topology. *C* executes a traceroute with destination *S*. The different stages of one of the packets and its reply traversing the network are shown as tables. Fill in the missing fields directly in Figure 4.1. The Protocol field should be the highest layer protocol of the packet. You can abbreviate Src MAC, Dst MAC, Src IP, and Dst IP as *device.interface*, for example $R_0.eth0$. Make meaningful choices for the values of the TTL and Protocol fields.

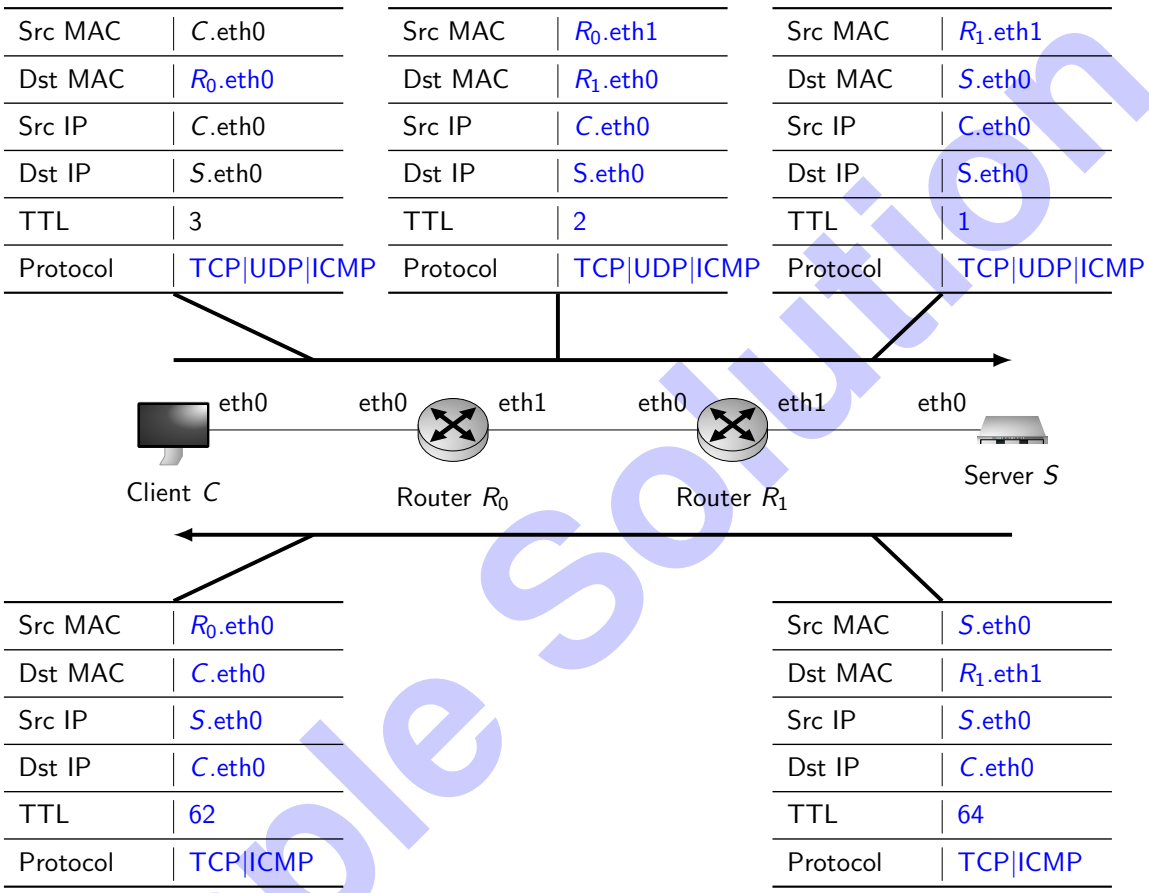

Figure 4.1: Network topology with packets between hops

Figure 4.2 shows a network topology. C executes a UDP-based traceroute with destination S. Figure 4.3 shows a hexdump of an ICMPv4 packet (starting with the first byte of the ICMPv4 header), which is one of the packets received by C as a result of the executed traceroute.

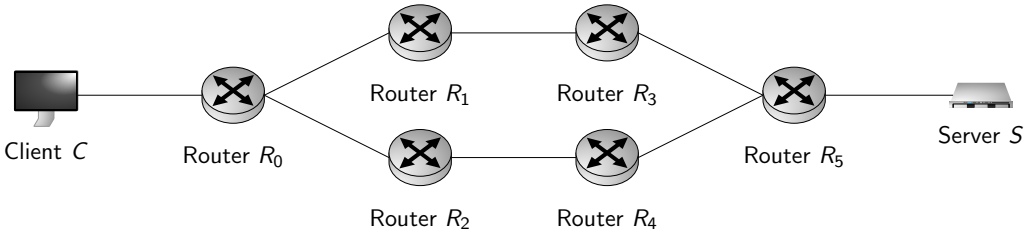

Figure 4.2: Network topology

```
0x0000  03 03 32 68 00 00 00 00    45 80 00 3c b2 f3 00 00
0x0010  01 11 c3 11 83 9f 14 2f    d8 3a d0 23 84 a8 82 bc
0x0020  00 28 c3 07
```

Figure 4.3: Hexdump of an ICMPv4 packet received at C

b)* Use the hexdump in Figure 4.3 to explain in detail which node from Figure 4.2 most likely originated the ICMPv4 packet and why.

0 1 2 3

- Type 3: Destination Unreachable
- Code 3: Destination Port Unreachable
- UDP-based traceroute sends packets to unlikely ports

Packet originated from S because a UDP traceroute sends to ports that are most likely closed and the correct ICMPv4 message to respond with in this case is Destination Unreachable, Destination Port Unreachable. Routers on the path would most likely respond with a Time Exceeded, TTL Expired in Transit.

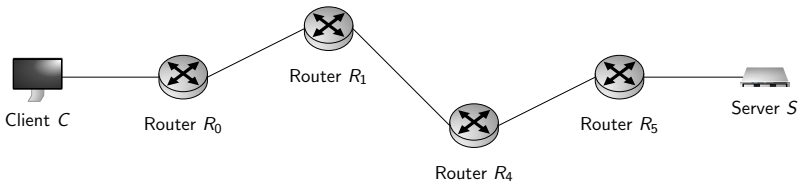

Figure 4.4: Erroneously detected network topology.

0 1 2 3 4

c) Explain a scenario where a traceroute on the network topology in Figure 4.2 erroneously detectes it to look like in Figure 4.4.

- R_0 acts as a load balancer
- Packets with TTL=2 are sent to R_1, packets expire at R_1
- Packets with TLL=3 are sent to R_2, packets expire at R_4
- Therefore, R_2 and R_3 are never detected and the wrong path between R_1 and R_4 is inferred

0 1

d)* Briefly explain how Paris-Traceroute works and what advantage it has over traditional traceroute.

- Vary header fields
- More precise detection of load balancing effects: missing nodes, wrong links

For the following subproblems, assume R_5 in Figure 4.2 blocks all types of ICMPv4 traffic and does not generate any type of ICMPv4 traffic itself. Assume traceroute sends exactly one packet for each TTL, and each hop introduces a **one way** latency of 10 ms.

0 1

e)* C executes an ICMPv4 traceroute with target S. Table 4.1 shows the output of the traceroute. Fill in the missing Round Trip Time (RTT) values.

0 1

f)* C executes a UDP traceroute with target S. Table 4.2 shows the output of the traceroute. Fill in the missing Round Trip Time (RTT) values.

Hop	IP address	RTT
1	R_0.ip	20 ms
2	R_1.ip	40 ms
3	R_3.ip	60 ms
4	R_5.ip	*
5	S.ip	*

Table 4.1: Output of ICMPv4 traceroute

Hop	IP address	RTT
1	R_0.ip	20 ms
2	R_1.ip	40 ms
3	R_3.ip	60 ms
4	R_5.ip	*
5	S.ip	*

Table 4.2: Output of UDP traceroute

Problem 5 Software-Defined Networking (13.5 credits)

This problem investigates a Software-Defined Network (SDN) powered by P4.

Plane	Description
Management plane	Configuration interface for administrator, e.g. for setting link costs
Control plane	Executes network algorithms, creating table entries for data plane
Forwarding/data plane	Forwarding of packets according to rule entries

Table 5.1: Planes of a Software-Defined Network

a)* List the three planes used in SDN in Table 5.1.

0
1

b) Briefly explain one of the main tasks of each plane in Table 5.1.

0
1

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. PCP and DEI are always set to `0`.

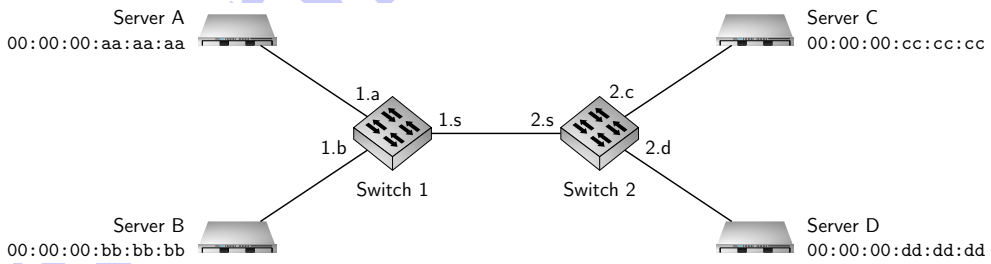

Figure 5.1: Network topology

c)* A VLAN capable switch has two different types of ports. Name the two different types and classify all ports of the network in Figure 5.1 (e.g. 1.a) in those two classes.

0
1
2

Port type	List of ports
Access ports (untagged)	1.a, 1.b, 2.c, 2.d
Trunk ports (tagged)	1.s, 2.s

Switches 1 and 2 are P4 switches handling the VLAN functionality. In the following the P4 program running on both switches is investigated. Listing 1 shows a small part of the used P4 program.

```
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) {
  // to be programmed
}

control Pipeline(inout headers hdr, inout metadata meta, inout standard_metadata_t std_meta) {
  action drop() {
    mark_to_drop();
  }
  action encap(bit<16> egress, bit<3> pcp, bit<1> dei, bit<12> vid) {
    std_meta.egress_port = egress;
    hdr.veth_ext.setValid();
    hdr.veth_ext.pcp = pcp;
    hdr.veth_ext.dei = dei;
    hdr.veth_ext.vid = vid;
  }
  table forward {
    actions = {
      encap;
      drop;
    }
    key = {
      std_meta.ingress_port: exact;
      hdr.eth.src: exact;
    }
    size = 4;
  }
  apply {
    if (hdr.veth_ext.isInvalid()) {
      forward.apply();
    }
  }
  // decapsulation functionality omitted
}
```

Listing 1: VLAN P4 program

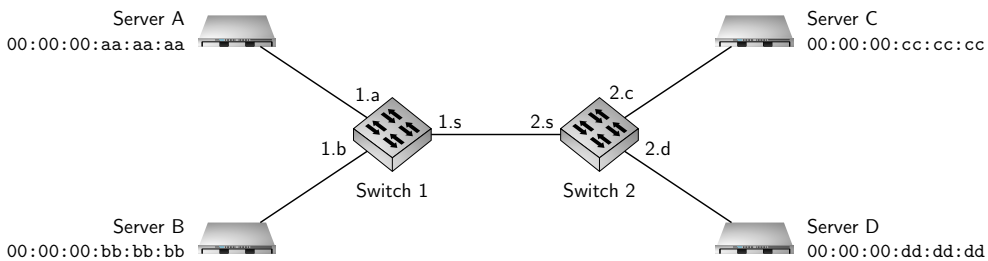

Figure 5.2: Network topology (Copy from previous page)

d)* Create a parser for the program given in Listing 1. The parser should be able to correctly parse VLAN and non-VLAN frames.

0 1 2 3

```
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: accept;
    }
  }
  state parse_veth_ext {
    packet.extract(hdr.veth_ext);
    transition accept;
  }
  state start {
    transition parse_eth;
  }
}
```

e)* The P4 program cannot work correctly without table data containing correct forwarding rules. Give the rules for Switch 1, to correctly encapsulate and forward frames of Clients A and B. Packets not originating from either A or B should be dropped. Use the information given in Figure 5.2. Use $*$ to mark the table cells which match on any value for which no more specific entry exists in this table.

0 1 2 3 4 5

Match field(s)	Key	Action	Action data
std_meta.ingress_port hdr.eth.src	1.a 00:00:00:aa:aa:aa	encap	egress=1.s pcp=0 dei=0 vid=16
std_meta.ingress_port hdr.eth.src	1.b 00:00:00:bb:bb:bb	encap	egress=1.s pcp=0 dei=0 vid=32
default	*	drop	*

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