Routerlab

Summer semester 2018

Worksheet 10
Group 08

Valentin Franck, 364066
Nikhil Singh, 387694

Pages: 25

Submission Date: July 12, 2018

Question 1

1a

The function of Openflow controller is to connect and configure the network devices such as switch, router etc by using openflow protocol to determine the best path for application traffic. The remote controller manages the switch's flow table (add, remove, modify entries). The switch's flow table is used by the switch to forward packets. Whenever no match is found, the packet is forwarded to the controller, which will then decide how to handle the packet.

1b

In SDN data plane and control plane are separated, allowing the use of dedicated hardware for each purpose, that can also be locally separated. The data plane contains the normal network traffic (from users). This traffic is forwarded by the switches. The traffic necessary to manage switches by controllers also passed by the same network and this traffic is part of the control plane.

1c

The flow table is used by the switch to handle packets, i.e. forward and drop them as specified in the table. For this purpose the table contains entries and matches header field against these entries. The flow table is modified by the controller.

1d

The entry can match against the Ethernet headers such as source and destination MAC and Ether type (but also against headers of the payload protocols such as IP and TCP source and destination and of course VLAN tags).

The payload (IP header, TCP/UDP header, TCP/UDP payload etc) is part of the Ethernet frame, so this question is not really very specific. For a full table of which entries can be matched, see Table 2 and 3 of the Openflow Specification.

1e

The following actions are supported by all Openflow-enabled switches: Forward, Drop, Enqueue, Modify-Field.

- If no action is specified, the packet is dropped.
- Enqueue forwards the packet through a queue which is attached to a port. This can be used for Quality of Service for example.
- Modify-Field is used to replace header fields of a packet for example the IP source address can be replaced.

1f

One example where OpenFlow can be useful is Intrusion Detection and Prevention. For instance all suspicious traffic might be forwarded to the controller by switches, so that there is a centralized instance, that can analyze traffic and then take action (by modifying the switches tables) if necessary.

OpenFlow can be used to make the setup of a network easier, as it allows to setup connectivity from the centralized controller. This means by using this abstraction (see SDN below) it gets easier, as we might simply specify which devices should be able to talk to which other devices and then the controller sets up VLANs, IP addresses, forwarding rules etc as needed.

1g

OpenFlow introduces a new kind of single point of failure. This is because the failure of a controller has severe consequences, as it affects a whole number of switches.

OpenFlow creates additional network traffic, which is necessary for the management of switches by controllers and especially if there is a high amount of packets without matching rules, that are forwarded to controllers, this will not scale.

1h

We could configure the switch such that it forwards all traffic from routing protocols to the controller. The controller can then process this traffic and modify the switch's table accordingly. The switch can determine the routing traffic based on IP addresses and port numbers. However, it is tricky because the controller might be in a different network and will have to make sure that certain traffic is actually routed via the same switch. The problem here might be that it is too slow if the switch has to forward a lot of messages back to the controller. Also switches unlike routers have ports and not interfaces which means they do not have IP addresses assigned to them (but as mentioned this problem can be solved by forwarding messages to the controller, that has assigned a virtual IP address to the switch used for routing).

One reason why this is not a proper router, is that is not possible to decrement the TTL at the the switch as it is not part of the modify-field actions set (therefore traceroute will not work). Also the router will not be able to recompute the checksum.

Question 2

2a

Mininet host run standard Linux, while switches support OpenFlow. It can only be used for OpenFlow and all devices are emulated in a single kernel. So it is limited in network size and bandwidth. Other network simulators like ns3 are not focused on OpenFlow.

Unlike other simulation solutions Mininet does not virtualize the full system as whole (and it does not need additional dedicated hardware.)

2b

Mininet uses process-based virtualization to run many hosts and switches in one kernel. This makes it efficient, because different proccesses can be handled in different CPU cores.

Mininet uses network namespaces. This allows each of the beforementioned virtual processes to get access to individually named network interfaces, routing tables, ARP tables etc. So this is another important virtualization feature.

2c

Mininet currently only supports OpenFlow switches. Mininet runs real coded (Linux kernel and network stack), which means it will usually also work in the real world.

2d

It supports 1.0 by default with Open vSwitch2.0.2. Version 1.3 can also be supported by adding "–switch ovs,protocols=OpenFlow13" from command line or passing "protocols='OpenFlow13'" to the OVSSwitch constructor.

2e

We need to make sure the host running Mininet is physically connected properly to the switch (via Ethernet) and also bring up the interface and assign it an IP address. We have to setup port-forwarding (port 6633) so that the guest VM running Mininet is able to use the interface

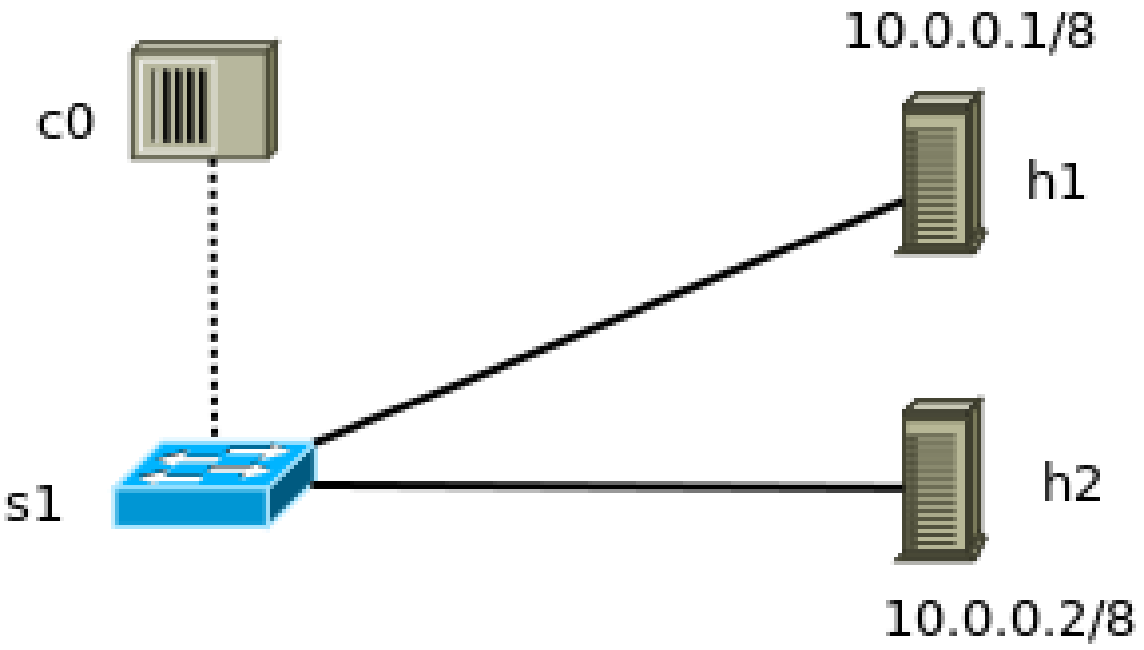

Figure 1: The default topology of Mininet. The dotted line means that there is no actual link set up between s1 and c0.

from the host and the incoming traffic will be forwarded to the Mininet controller. Once both the switch is set up and the host and VM are configured, we should be able to have controller and switch talk to each other.

Question 3

3a

For the topology see Figure: 1.

When we use "sudo mn" it first creates network then adds controller c0 then add hosts which are h1 and h2 here then add switch which is s1. After this it add links (h1, s1) (h2, s1) then it configure hosts h1 and h2 then it starts controller c0 and then starts switch s1 and then starts CLI. To check the IP address we ran this command "h1 ifconfig -a" which shows IP 10.0.0.1 with mask 255.0.0.0 is assigned to h1 on eth0. Then we ran "h2 ifconfig -a" which shows IP 10.0.0.2 with mask 255.0.0.0 is assigned to h2 on eth0. The default topology is started. It contains one OpenFlow kernel switch connected to two hosts, plus the OpenFlow reference controller.

```
mininet@mininet-vm:~$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
```

```
s1 ...
*** Starting CLI:
mininet>
```

3b

OpenFlow controller uses port number 6653 and the loopback interface. However OpenVSwitch uses None. It also uses the loopback interface. We see two network interfaces from h1 namespace. First is "h1-eth0" and second is "lo" which is the loopback interface.

```
mininet> h1 ifconfig
h1-eth0   Link encap:Ethernet  HWaddr 82:ac:59:d2:4c:c7
          inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2367>
<Host h2: h2-eth0:10.0.0.2 pid=2369>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2374>
<Controller c0: 127.0.0.1:6653 pid=2360>
mininet> c0 netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
   State
tcp        0      0 192.168.56.101:ssh      192.168.56.1:58512
   ESTABLISHED
tcp        0      0 localhost:46728         localhost:6653
   ESTABLISHED
tcp        0      0 localhost:6653          localhost:46728
   ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    17231    /tmp/vlogs
   .2420
unix  13     [ ]         DGRAM                    8621     /dev/log
unix  3      [ ]         STREAM     CONNECTED     9427     /var/run/
   openvswitch/db.sock
unix  3      [ ]         STREAM     CONNECTED     10830    /var/run/
   dbus/system_bus_socket
unix  3      [ ]         STREAM     CONNECTED     8909
unix  2      [ ]         DGRAM                    9418
unix  2      [ ]         STREAM     CONNECTED     17046
unix  2      [ ]         STREAM     CONNECTED     13840
unix  2      [ ]         DGRAM                    9386
unix  2      [ ]         DGRAM                    17229
```

```
unix  3      [ ]         STREAM     CONNECTED     8646     /var/run/
    dbus/system_bus_socket
unix  2      [ ]         DGRAM                    8720
unix  2      [ ]         DGRAM                    13842
unix  3      [ ]         DGRAM                    8376
unix  2      [ ]         DGRAM                    10552
unix  2      [ ]         DGRAM                    17048
unix  3      [ ]         STREAM     CONNECTED     8587     @/com/ubuntu
    /upstart
unix  3      [ ]         STREAM     CONNECTED     8548
unix  3      [ ]         STREAM     CONNECTED     8370     @/com/ubuntu
    /upstart
unix  3      [ ]         STREAM     CONNECTED     10821
unix  3      [ ]         STREAM     CONNECTED     10829
unix  3      [ ]         STREAM     CONNECTED     8575
unix  2      [ ]         DGRAM                    9969
unix  3      [ ]         STREAM     CONNECTED     8360
unix  3      [ ]         STREAM     CONNECTED     8946     @/com/ubuntu
    /upstart
unix  3      [ ]         STREAM     CONNECTED     8576
unix  3      [ ]         DGRAM                    8377
unix  3      [ ]         STREAM     CONNECTED     8588
unix  2      [ ]         DGRAM                    8717
unix  3      [ ]         STREAM     CONNECTED     8645
unix  3      [ ]         STREAM     CONNECTED     9425
unix  2      [ ]         DGRAM                    8622
unix  3      [ ]         STREAM     CONNECTED     10822
unix  2      [ ]         DGRAM                    10727
unix  3      [ ]         STREAM     CONNECTED     8589     /var/run/
    dbus/system_bus_socket
```

3c

The command "sh ovs-ofctl dump-flows s1" If we are running OVS we can pass the switch name to ovs-ofctl which connects to it via file system and it dumps all the flows on switch s1. We see the response as

```
"NXST_FLOW reply (xid=0x4):"
```

It means their is no traffic flow at the moment on s1. We pinged h2 from h1 once by using this command "h1 ping -c 1 h2" and then ran this command again "sh ovs-ofctl dump-flows s1" and this time we received the dump of the flow on s1. We can see that we have a bidirectional rule which allows communication between packets from h1 to h2 and vice-versa which is defined by the mac addresses. dl-src and dl-dst which you could see in the output provided. The main things to notice here are "duration" which refers to the number of seconds the entry is in the table. "Table" which refers to the specific table in which flow is installed which in our case is table =0. "n-packets" which refers to number of packets which have matched the entry which is 1 in our case. "idle-age" which refers to the number of seconds since the last packet matched the entry.

```
mininet> sh ovs-ofctl dump-flows s1
NXST_FLOW reply (xid=0x4):
mininet> h1 ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=5.07 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.01 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.084 ms
```

```
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.081 ms
^C
--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3002ms
rtt min/avg/max/mdev = 0.081/1.562/5.075/2.063 ms
mininet> sh ovs-ofctl dump-flows s1
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=4.117s, table=0, n_packets=1, n_bytes=42,
     idle_timeout=60, idle_age=4, priority=65535,arp,in_port=2,vlan_tci
     =0x0000,dl_src=4a:5b:a7:a7:bf:c1,dl_dst=de:00:0e:12:9e:2f,arp_spa
     =10.0.0.2,arp_tpa=10.0.0.1,arp_op=1 actions=output:1
 cookie=0x0, duration=4.114s, table=0, n_packets=1, n_bytes=42,
     idle_timeout=60, idle_age=4, priority=65535,arp,in_port=1,vlan_tci
     =0x0000,dl_src=de:00:0e:12:9e:2f,dl_dst=4a:5b:a7:a7:bf:c1,arp_spa
     =10.0.0.1,arp_tpa=10.0.0.2,arp_op=2 actions=output:2
 cookie=0x0, duration=9.121s, table=0, n_packets=1, n_bytes=42,
     idle_timeout=60, idle_age=9, priority=65535,arp,in_port=2,vlan_tci
     =0x0000,dl_src=4a:5b:a7:a7:bf:c1,dl_dst=de:00:0e:12:9e:2f,arp_spa
     =10.0.0.2,arp_tpa=10.0.0.1,arp_op=2 actions=output:1
 cookie=0x0, duration=9.12s, table=0, n_packets=4, n_bytes=392,
     idle_timeout=60, idle_age=6, priority=65535,icmp,in_port=1,
     vlan_tci=0x0000,dl_src=de:00:0e:12:9e:2f,dl_dst=4a:5b:a7:a7:bf:c1,
     nw_src=10.0.0.1,nw_dst=10.0.0.2,nw_tos=0,icmp_type=8,icmp_code=0
     actions=output:2
 cookie=0x0, duration=9.118s, table=0, n_packets=4, n_bytes=392,
     idle_timeout=60, idle_age=6, priority=65535,icmp,in_port=2,
     vlan_tci=0x0000,dl_src=4a:5b:a7:a7:bf:c1,dl_dst=de:00:0e:12:9e:2f,
     nw_src=10.0.0.2,nw_dst=10.0.0.1,nw_tos=0,icmp_type=0,icmp_code=0
     actions=output:1
```

3d

We already answered that in 3c.

3e

This command "sudo mn –co none" does everything as "sudo mn" except id doesn't add c0 controller to the topology because we used –co none. When we pinged h2 from h1 the packets got lost which means it couldn't ping because their is no controller. We need to add the controller to the topology first and then it will ping successfully. This is because otherwise the switch's table is empty and it will try to forward the packet to a controller, which does not exist. So the packets are dropped by the switch. This does not happen if the controller can handle the traffic and adjust the switch's table.

```
mininet> h1 ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
^C
--- 10.0.0.2 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time
    4024ms
pipe 3
```

Question 4

4a

Here is how we captured the traffic. Note that we changed the port, because it does not make sense to capture from port 6633, when there is absolutely no traffic/ process running on it:

```
mininet> c0 tcpdump -i lo -s 65535 -w my_dump port 6653 &
mininet> h1 ping 10.0.0.2 -c 1
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=5.84 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.841/5.841/5.841/0.000 ms
mininet> c0 ps
tcpdump: listening on lo, link-type EN10MB (Ethernet), capture size
    262144 bytes
  PID TTY          TIME CMD
 4319 pts/11   00:00:00 bash
 4381 pts/11   00:00:00 controller
 4444 pts/11   00:00:00 tcpdump
 4449 pts/11   00:00:00 ps
mininet> c0 kill 4444
22 packets captured
44 packets received by filter
0 packets dropped by kernel
```

Here you can see and overview of all captured packets:

```
mininet@mininet-vm:~$ tshark -r my_dump
  1    0.000000     127.0.0.1 -> 127.0.0.1     OF 1.0 74 of_echo_request
  2    0.002358     127.0.0.1 -> 127.0.0.1     OF 1.0 74 of_echo_reply
  3    0.002393     127.0.0.1 -> 127.0.0.1     TCP 66 46762 > openflow [
     ACK] Seq=9 Ack=9 Win=86 Len=0 TSval=1233500 TSecr=1233500
  4    2.249621 0e:32:28:e1:b8:da -> Broadcast    OF 1.0 126
     of_packet_in
  5    2.250355     127.0.0.1 -> 127.0.0.1     OF 1.0 90 of_packet_out
  6    2.250375     127.0.0.1 -> 127.0.0.1     TCP 66 46762 > openflow [
     ACK] Seq=69 Ack=33 Win=86 Len=0 TSval=1234062 TSecr=1234062
  7    2.250747 16:e4:44:53:ad:4b -> 0e:32:28:e1:b8:da OF 1.0 126
     of_packet_in
  8    2.251317     127.0.0.1 -> 127.0.0.1     OF 1.0 146 of_flow_add
  9    2.252065      10.0.0.1 -> 10.0.0.2      OF 1.0 182 of_packet_in
 10    2.252589     127.0.0.1 -> 127.0.0.1     OF 1.0 146 of_flow_add
 11    2.253042      10.0.0.2 -> 10.0.0.1      OF 1.0 182 of_packet_in
 12    2.253477     127.0.0.1 -> 127.0.0.1     OF 1.0 146 of_flow_add
 13    2.294831     127.0.0.1 -> 127.0.0.1     TCP 66 46762 > openflow [
    ACK] Seq=361 Ack=273 Win=86 Len=0 TSval=1234073 TSecr=1234063
 14    7.000342     127.0.0.1 -> 127.0.0.1     OF 1.0 74 of_echo_request
 15    7.000697     127.0.0.1 -> 127.0.0.1     OF 1.0 74 of_echo_reply
 16    7.000718     127.0.0.1 -> 127.0.0.1     TCP 66 46762 > openflow [
    ACK] Seq=369 Ack=281 Win=86 Len=0 TSval=1235250 TSecr=1235250
 17    7.264540 16:e4:44:53:ad:4b -> 0e:32:28:e1:b8:da OF 1.0 126
     of_packet_in
 18    7.265101     127.0.0.1 -> 127.0.0.1     OF 1.0 146 of_flow_add
 19    7.265130     127.0.0.1 -> 127.0.0.1     TCP 66 46762 > openflow [
    ACK] Seq=429 Ack=361 Win=86 Len=0 TSval=1235316 TSecr=1235316
```

```
 20    7.266817 0e:32:28:e1:b8:da -> 16:e4:44:53:ad:4b OF 1.0 126
    of_packet_in
 21    7.267188     127.0.0.1 -> 127.0.0.1     OF 1.0 146 of_flow_add
 22    7.303607     127.0.0.1 -> 127.0.0.1     TCP 66 46762 > openflow [
    ACK] Seq=489 Ack=441 Win=86 Len=0 TSval=1235326 TSecr=1235316
```

And here is the full output from the first 4 frames (of 22). For the rest see the provided dump_q4.pcap file:

```
mininet@mininet-vm:~$ tshark -V -r my_dump
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
    Encapsulation type: Ethernet (1)
    Arrival Time: Jul 10, 2018 10:50:20.101130000 PDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1531245020.101130000 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 74 bytes (592 bits)
    Capture Length: 74 bytes (592 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ip:tcp:of]
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00
    _00:00:00 (00:00:00:00:00:00)
    Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
            unicast)
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
            unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1
    (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6;
        ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
        1100 00.. = Differentiated Services Codepoint: Class Selector 6
            (0x30)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-
            Capable Transport) (0x00)
    Total Length: 60
    Identification: 0xc98d (51597)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
```

```
    Protocol: TCP (6)
    Header checksum: 0x726c [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 46762 (46762), Dst Port:
   openflow (6653), Seq: 1, Ack: 1, Len: 8
    Source port: 46762 (46762)
    Destination port: openflow (6653)
    [Stream index: 0]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 9    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Header length: 32 bytes
    Flags: 0x018 (PSH, ACK)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...1 .... = Acknowledgment: Set
        .... .... 1... = Push: Set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
    Window size value: 86
    [Calculated window size: 86]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xfe30 [validation disabled]
        [Good Checksum: False]
        [Bad Checksum: False]
    Options: (12 bytes), No-Operation (NOP), No-Operation (NOP),
       Timestamps
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        Timestamps: TSval 1233500, TSecr 1232250
            Kind: Timestamp (8)
            Length: 10
            Timestamp value: 1233500
            Timestamp echo reply: 1232250
    [SEQ/ACK analysis]
        [Bytes in flight: 8]
OpenFlow (LOXI)
    version: 1
    type: OFPT_ECHO_REQUEST (2)
```

```
    length: 8
    xid: 0

Frame 2: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
    Encapsulation type: Ethernet (1)
    Arrival Time: Jul 10, 2018 10:50:20.103488000 PDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1531245020.103488000 seconds
    [Time delta from previous captured frame: 0.002358000 seconds]
    [Time delta from previous displayed frame: 0.002358000 seconds]
    [Time since reference or first frame: 0.002358000 seconds]
    Frame Number: 2
    Frame Length: 74 bytes (592 bits)
    Capture Length: 74 bytes (592 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ip:tcp:of]
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00
    _00:00:00 (00:00:00:00:00:00)
    Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
            unicast)
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
            unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1
    (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00:
        Not-ECT (Not ECN-Capable Transport))
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-
            Capable Transport) (0x00)
    Total Length: 60
    Identification: 0xc376 (50038)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x7943 [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
    [Source GeoIP: Unknown]
```

```
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: openflow (6653), Dst Port:
    46762 (46762), Seq: 1, Ack: 9, Len: 8
    Source port: openflow (6653)
    Destination port: 46762 (46762)
    [Stream index: 0]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 9    (relative sequence number)]
    Acknowledgment number: 9    (relative ack number)
    Header length: 32 bytes
    Flags: 0x018 (PSH, ACK)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...1 .... = Acknowledgment: Set
        .... .... 1... = Push: Set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
    Window size value: 88
    [Calculated window size: 88]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xfe30 [validation disabled]
        [Good Checksum: False]
        [Bad Checksum: False]
    Options: (12 bytes), No-Operation (NOP), No-Operation (NOP),
        Timestamps
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        Timestamps: TSval 1233500, TSecr 1233500
            Kind: Timestamp (8)
            Length: 10
            Timestamp value: 1233500
            Timestamp echo reply: 1233500
    [SEQ/ACK analysis]
        [This is an ACK to the segment in frame: 1]
        [The RTT to ACK the segment was: 0.002358000 seconds]
        [Bytes in flight: 8]
OpenFlow (LOXI)
    version: 1
    type: OFPT_ECHO_REPLY (3)
    length: 8
    xid: 0

Frame 3: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
    Encapsulation type: Ethernet (1)
```

```
    Arrival Time: Jul 10, 2018 10:50:20.103523000 PDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1531245020.103523000 seconds
    [Time delta from previous captured frame: 0.000035000 seconds]
    [Time delta from previous displayed frame: 0.000035000 seconds]
    [Time since reference or first frame: 0.002393000 seconds]
    Frame Number: 3
    Frame Length: 66 bytes (528 bits)
    Capture Length: 66 bytes (528 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ip:tcp]
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00
    _00:00:00 (00:00:00:00:00:00)
    Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
           unicast)
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
           unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1
    (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6;
       ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
        1100 00.. = Differentiated Services Codepoint: Class Selector 6
            (0x30)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-
           Capable Transport) (0x00)
    Total Length: 52
    Identification: 0xc98e (51598)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x7273 [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 46762 (46762), Dst Port:
   openflow (6653), Seq: 9, Ack: 9, Len: 0
    Source port: 46762 (46762)
```

```
    Destination port: openflow (6653)
    [Stream index: 0]
    Sequence number: 9    (relative sequence number)
    Acknowledgment number: 9    (relative ack number)
    Header length: 32 bytes
    Flags: 0x010 (ACK)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...1 .... = Acknowledgment: Set
        .... .... 0... = Push: Not set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
    Window size value: 86
    [Calculated window size: 86]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xfe28 [validation disabled]
        [Good Checksum: False]
        [Bad Checksum: False]
    Options: (12 bytes), No-Operation (NOP), No-Operation (NOP),
       Timestamps
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        Timestamps: TSval 1233500, TSecr 1233500
            Kind: Timestamp (8)
            Length: 10
            Timestamp value: 1233500
            Timestamp echo reply: 1233500
    [SEQ/ACK analysis]
        [This is an ACK to the segment in frame: 2]
        [The RTT to ACK the segment was: 0.000035000 seconds]

Frame 4: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits)
    Encapsulation type: Ethernet (1)
    Arrival Time: Jul 10, 2018 10:50:22.350751000 PDT
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1531245022.350751000 seconds
    [Time delta from previous captured frame: 2.247228000 seconds]
    [Time delta from previous displayed frame: 2.247228000 seconds]
    [Time since reference or first frame: 2.249621000 seconds]
    Frame Number: 4
    Frame Length: 126 bytes (1008 bits)
    Capture Length: 126 bytes (1008 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
```

```
    [Protocols in frame: eth:ip:tcp:of:eth:arp]
Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00
    _00:00:00 (00:00:00:00:00:00)
    Destination: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
            unicast)
    Source: 00:00:00_00:00:00 (00:00:00:00:00:00)
        Address: 00:00:00_00:00:00 (00:00:00:00:00:00)
        .... ..0. .... .... .... .... = LG bit: Globally unique address
            (factory default)
        .... ...0 .... .... .... .... = IG bit: Individual address (
            unicast)
    Type: IP (0x0800)
Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1
    (127.0.0.1)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0xc0 (DSCP 0x30: Class Selector 6;
        ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
        1100 00.. = Differentiated Services Codepoint: Class Selector 6
            (0x30)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-
            Capable Transport) (0x00)
    Total Length: 112
    Identification: 0xc98f (51599)
    Flags: 0x02 (Don't Fragment)
        0... .... = Reserved bit: Not set
        .1.. .... = Don't fragment: Set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x7236 [validation disabled]
        [Good: False]
        [Bad: False]
    Source: 127.0.0.1 (127.0.0.1)
    Destination: 127.0.0.1 (127.0.0.1)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Transmission Control Protocol, Src Port: 46762 (46762), Dst Port:
    openflow (6653), Seq: 9, Ack: 9, Len: 60
    Source port: 46762 (46762)
    Destination port: openflow (6653)
    [Stream index: 0]
    Sequence number: 9    (relative sequence number)
    [Next sequence number: 69    (relative sequence number)]
    Acknowledgment number: 9    (relative ack number)
    Header length: 32 bytes
    Flags: 0x018 (PSH, ACK)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
```

```
        .... ..0. .... = Urgent: Not set
        .... ...1 .... = Acknowledgment: Set
        .... .... 1... = Push: Set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
    Window size value: 86
    [Calculated window size: 86]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0xfe64 [validation disabled]
        [Good Checksum: False]
        [Bad Checksum: False]
    Options: (12 bytes), No-Operation (NOP), No-Operation (NOP),
        Timestamps
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        No-Operation (NOP)
            Type: 1
                0... .... = Copy on fragmentation: No
                .00. .... = Class: Control (0)
                ...0 0001 = Number: No-Operation (NOP) (1)
        Timestamps: TSval 1234062, TSecr 1233500
            Kind: Timestamp (8)
            Length: 10
            Timestamp value: 1234062
            Timestamp echo reply: 1233500
    [SEQ/ACK analysis]
        [Bytes in flight: 60]
OpenFlow (LOXI)
    version: 1
    type: OFPT_PACKET_IN (10)
    length: 60
    xid: 0
    buffer_id: 256
    total_len: 42
    in_port: 1
    reason: OFPR_NO_MATCH (0)
    Ethernet packet
        Ethernet II, Src: 0e:32:28:e1:b8:da (0e:32:28:e1:b8:da), Dst:
            Broadcast (ff:ff:ff:ff:ff:ff)
            Destination: Broadcast (ff:ff:ff:ff:ff:ff)
                Address: Broadcast (ff:ff:ff:ff:ff:ff)
                .... ..1. .... .... .... .... = LG bit: Locally
                    administered address (this is NOT the factory
                    default)
                .... ...1 .... .... .... .... = IG bit: Group address (
                    multicast/broadcast)
            Source: 0e:32:28:e1:b8:da (0e:32:28:e1:b8:da)
                Address: 0e:32:28:e1:b8:da (0e:32:28:e1:b8:da)
                .... ..1. .... .... .... .... = LG bit: Locally
                    administered address (this is NOT the factory
                    default)
                .... ...0 .... .... .... .... = IG bit: Individual
```

```
            address (unicast)
        Type: ARP (0x0806)
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: 0e:32:28:e1:b8:da (0e:32:28:e1:b8:da)
    Sender IP address: 10.0.0.1 (10.0.0.1)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 10.0.0.2 (10.0.0.2)
```

The first two messages are the icmp request reply messages from the ping and the third is the respective ACK. We can see that packet 4 has broadcast as destination. That is because it is the initial message of ARP in order to find the MAC address corresponding to the given IP address. We can also see several flow modifications which the controller sets at the switch.

Question 5

5a

The relationship between SDN and OpenFlow is SDN has NOS which is network operating system which conveys configuration of global network view to real physical devices and OpenFlow is a forwarding protocol which interacts between data plane and control plane. OpenFlow plays a role and is one possible solution to model configuration of the physical device.

5b

Two motivating factors behind SDN are the future of networking by making the abstractions cleaner instead of complicated distributed protocols and to make networking a mature discipline by extracting simplicity from the sea of complexity.

5c

Three different benefits of SDN would be easy to configure, easy to monitor traffic, no routing protocols necessary. For example for network administrators it is possible to just configure the controller, which will then apply the resulting rules to the individual devices without the need to configure all devices individually. This may include assigning IP addresses, setting up VLANs and configuring firewall and forwarding rules just by using an abstraction (without the need to implement each step individually).

The centralized view from the controller can provide easy debugging, because it is easier to (automatically) detect misconfigurations.

When it comes to traffic shaping and also to intrusion detection the cnetralized view comes in handy, because it allows for quick adaptation of the rules in the network.

5d

What Professor Shenker means is that SDN is an example of fundamental abstractions which were needed to alienate concerns to make the networking issues easy to control and manageable in a simplistic way. SDN is the right abstraction because it draws attention to the right things. So what he means is, that SDN is rather about the question, what you want to happen in the network than about the question how you implement that.

Professor Shenker mentioned that OpenFlow, ONIX etc are the implementations which might change over time (because they are not necessarily the right answers, but only what works now). Nevertheless, abstractions will remain the same which could make us reliably build much more complicated functionality.

An example of key abstraction that SDN builds upon are the creation of reusable and easy to deal with building blocks such as Forwarding path, Network operating system and Nypervisor which is the Hypervisor of network. These allow SDN to use abstractions such Global Management abstraction (manage the network from a single controller using the Nypervisor), Network View Abstraction (NOS only shows what is relevant for configuration to Nypervisor without providing all the details of the network and the view does not have to be correct at once but it will be correct eventually as it adapts), and Forwarding Interface Abstraction (used by the NOS to configure the actual physical hardware that does the forwarding).

Question 6

Because the delay was quite long so first we ran this command for Hub

```
mininet@mininet-vm:~/pox$ ./pox.py log.level --DEBUG forwarding.hub
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:forwarding.hub:Hub running.
DEBUG:core:POX 0.2.0 (carp) going up...
DEBUG:core:Running on CPython (2.7.6/Oct 26 2016 20:30:19)
DEBUG:core:Platform is Linux-4.2.0-27-generic-x86_64-with-Ubuntu-14.04-
    trusty
INFO:core:POX 0.2.0 (carp) is up.
DEBUG:openflow.of_01:Listening on 0.0.0.0:6633
INFO:openflow.of_01:[00-00-00-00-00-01 1] connected
INFO:forwarding.hub:Hubifying 00-00-00-00-00-01
```

For switch-

```
mininet@mininet-vm:~/pox$ ./pox.py forwarding.l2_learning
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[None 1] closed
INFO:openflow.of_01:[00-00-00-00-00-01 2] connected
```

and then pinged the hosts to confirm the connectivity

```
mininet@mininet-vm:~$ sudo mn --topo single,3 --mac --switch ovsk --
    controller remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Connecting to remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> h2 ping -c5 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=54.0 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.397 ms
```

```
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.058 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.036 ms
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=0.046 ms

--- 10.0.0.3 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4004ms
rtt min/avg/max/mdev = 0.036/10.921/54.068/21.573 ms
mininet> h1 ping -c3 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=28.0 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.522 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.055 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 0.055/9.556/28.092/13.108 ms
mininet> h1 ping -c3 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=22.2 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.577 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.523 ms

--- 10.0.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.523/7.769/22.207/10.209 ms
mininet> h2 ping -c3 h1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=15.9 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.371 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.060 ms

--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.060/5.456/15.937/7.412 ms
mininet> h3 ping -c3 h1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=18.7 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.489 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.060 ms

--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.060/6.445/18.786/8.728 ms
mininet> h3 ping -c3 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=32.3 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.339 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.059 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.059/10.901/32.307/15.136 ms
mininet>


mininet> pingall
```

```
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (6/6 received)
```

Verification of Hub behavior with tcpdump- First we pinged the host2 from h1 and got an identical response on h2 and h3

```
root@mininet-vm:~# ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.337 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.337/0.337/0.337/0.000 ms

root@mininet-vm:~# tcpdump -XX -n -i h3-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on h3-eth0, link-type EN10MB (Ethernet), capture size 262144
    bytes
08:38:48.926458 IP 10.0.0.1 > 10.0.0.2: ICMP echo request, id 7892, seq
     1, length 64
        0x0000:  0000 0000 0002 0000 0000 0001 0800 4500
            ..............E.
        0x0010:  0054 bc10 4000 4001 6a96 0a00 0001 0a00  .T..@.@.j
            .......
        0x0020:  0002 0800 7864 1ed4 0001 0876 475b 0000  ....xd.....vG
            [..
        0x0030:  0000 4422 0e00 0000 0000 1011 1213 1415  ..D
            ".............
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425
            ...........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()
            *+,-./012345
        0x0060:  3637                                     67
08:38:48.926612 IP 10.0.0.2 > 10.0.0.1: ICMP echo reply, id 7892, seq
    1, length 64
        0x0000:  0000 0000 0001 0000 0000 0002 0800 4500
            ..............E.
        0x0010:  0054 92ca 0000 4001 d3dc 0a00 0002 0a00  .T....@
            ........
        0x0020:  0001 0000 8064 1ed4 0001 0876 475b 0000  .....d.....vG
            [..
        0x0030:  0000 4422 0e00 0000 0000 1011 1213 1415  ..D
            ".............
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425
            ...........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()
            *+,-./012345
        0x0060:  3637                                     67
08:38:53.927673 ARP, Request who-has 10.0.0.2 tell 10.0.0.1, length 28
        0x0000:  0000 0000 0002 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
```

```
        0x0020:  0000 0000 0000 0a00 0002                  ..........
08:38:53.927709 ARP, Request who-has 10.0.0.1 tell 10.0.0.2, length 28
        0x0000:  0000 0000 0001 0000 0000 0002 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0002 0a00 0002
            ................
        0x0020:  0000 0000 0000 0a00 0001                  ..........
08:38:53.927883 ARP, Reply 10.0.0.1 is-at 00:00:00:00:00:01, length 28
        0x0000:  0000 0000 0002 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0002 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0002 0a00 0002                  ..........
08:38:53.927896 ARP, Reply 10.0.0.2 is-at 00:00:00:00:00:02, length 28
        0x0000:  0000 0000 0001 0000 0000 0002 0806 0001
            ................
        0x0010:  0800 0604 0002 0000 0000 0002 0a00 0002
            ................
        0x0020:  0000 0000 0001 0a00 0001                  ..........
```

```
root@mininet-vm:~# tcpdump -XX -n -i h2-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on h2-eth0, link-type EN10MB (Ethernet), capture size 262144
    bytes
08:38:48.926459 IP 10.0.0.1 > 10.0.0.2: ICMP echo request, id 7892, seq
     1, length 64
        0x0000:  0000 0000 0002 0000 0000 0001 0800 4500
            ..............E.
        0x0010:  0054 bc10 4000 4001 6a96 0a00 0001 0a00   .T..@.@.j
            .......
        0x0020:  0002 0800 7864 1ed4 0001 0876 475b 0000   ....xd.....vG
            [..
        0x0030:  0000 4422 0e00 0000 0000 1011 1213 1415   ..D
            "...........
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425
            ..........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435   &'()
            *+,-./012345
        0x0060:  3637                                      67
08:38:48.926475 IP 10.0.0.2 > 10.0.0.1: ICMP echo reply, id 7892, seq
    1, length 64
        0x0000:  0000 0000 0001 0000 0000 0002 0800 4500
            ..............E.
        0x0010:  0054 92ca 0000 4001 d3dc 0a00 0002 0a00   .T....@
            ........
        0x0020:  0001 0000 8064 1ed4 0001 0876 475b 0000   .....d.....vG
            [..
        0x0030:  0000 4422 0e00 0000 0000 1011 1213 1415   ..D
            "...........
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425
```

```
            ..........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()
            *+,-./012345
        0x0060:  3637                                     67
08:38:53.927547 ARP, Request who-has 10.0.0.1 tell 10.0.0.2, length 28
        0x0000:  0000 0000 0001 0000 0000 0002 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0002 0a00 0002
            ................
        0x0020:  0000 0000 0000 0a00 0001                 ..........
08:38:53.927674 ARP, Request who-has 10.0.0.2 tell 10.0.0.1, length 28
        0x0000:  0000 0000 0002 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0002                 ..........
08:38:53.927689 ARP, Reply 10.0.0.2 is-at 00:00:00:00:00:02, length 28
        0x0000:  0000 0000 0001 0000 0000 0002 0806 0001
            ................
        0x0010:  0800 0604 0002 0000 0000 0002 0a00 0002
            ................
        0x0020:  0000 0000 0001 0a00 0001                 ..........
08:38:53.927886 ARP, Reply 10.0.0.1 is-at 00:00:00:00:00:01, length 28
        0x0000:  0000 0000 0002 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0002 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0002 0a00 0002                 ..........
```

Then we pinged the non-existent host-

```
root@mininet-vm:~# ping -c 1 10.0.0.5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable

--- 10.0.0.5 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0
    ms

root@mininet-vm:~# tcpdump -XX -n -i h3-eth0
08:44:30.001655 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
        0x0000:  ffff ffff ffff 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0005                 ..........
08:44:30.999675 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
        0x0000:  ffff ffff ffff 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0005                 ..........
08:44:31.999324 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
        0x0000:  ffff ffff ffff 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
```

```
        0x0020:  0000 0000 0000 0a00 0005                  ..........


root@mininet-vm:~# tcpdump -XX -n -i h2-eth0
08:44:30.001656 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
        0x0000:  ffff ffff ffff 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0005                  ..........
08:44:30.999678 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
        0x0000:  ffff ffff ffff 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0005                  ..........
08:44:31.999326 ARP, Request who-has 10.0.0.5 tell 10.0.0.1, length 28
        0x0000:  ffff ffff ffff 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0005                  ..........
```

and observed the typical hub behavior because it flooded all the hosts and they could see the exact same traffic. Verification of Switch behavior with tcpdump-

```
root@mininet-vm:~# ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=2.90 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 2.902/2.902/2.902/0.000 ms



root@mininet-vm:~# tcpdump -XX -n -i h2-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on h2-eth0, link-type EN10MB (Ethernet), capture size 262144
    bytes
08:52:20.873634 IP 10.0.0.1 > 10.0.0.2: ICMP echo request, id 8344, seq
     1, length 64
        0x0000:  0000 0000 0002 0000 0000 0001 0800 4500
            ..............E.
        0x0010:  0054 99e9 4000 4001 8cbd 0a00 0001 0a00  .T..@.@
            ........
        0x0020:  0002 0800 966e 2098 0001 3479 475b 0000  .....n....4yG
            [..
        0x0030:  0000 f950 0d00 0000 0000 1011 1213 1415  ...P
            ............
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425
            ..........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()
            *+,-./012345
        0x0060:  3637                                     67
08:52:20.873649 IP 10.0.0.2 > 10.0.0.1: ICMP echo reply, id 8344, seq
```

```
    1, length 64
        0x0000:  0000 0000 0001 0000 0000 0002 0800 4500
            ..............E.
        0x0010:  0054 285f 0000 4001 3e48 0a00 0002 0a00  .T(_..@.>H
            ......
        0x0020:  0001 0000 9e6e 2098 0001 3479 475b 0000  .....n....4yG
            [..
        0x0030:  0000 f950 0d00 0000 0000 1011 1213 1415  ...P
            ............
        0x0040:  1617 1819 1a1b 1c1d 1e1f 2021 2223 2425
            ..........!"#$%
        0x0050:  2627 2829 2a2b 2c2d 2e2f 3031 3233 3435  &'()
           *+,-./012345
        0x0060:  3637                                     67
08:52:25.879434 ARP, Request who-has 10.0.0.1 tell 10.0.0.2, length 28
        0x0000:  0000 0000 0001 0000 0000 0002 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0002 0a00 0002
            ................
        0x0020:  0000 0000 0000 0a00 0001                 ..........
08:52:25.887785 ARP, Request who-has 10.0.0.2 tell 10.0.0.1, length 28
        0x0000:  0000 0000 0002 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0001 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0000 0a00 0002                 ..........
08:52:25.887805 ARP, Reply 10.0.0.2 is-at 00:00:00:00:00:02, length 28
        0x0000:  0000 0000 0001 0000 0000 0002 0806 0001
            ................
        0x0010:  0800 0604 0002 0000 0000 0002 0a00 0002
            ................
        0x0020:  0000 0000 0001 0a00 0001                 ..........
08:52:25.923468 ARP, Reply 10.0.0.1 is-at 00:00:00:00:00:01, length 28
        0x0000:  0000 0000 0002 0000 0000 0001 0806 0001
            ................
        0x0010:  0800 0604 0002 0000 0000 0001 0a00 0001
            ................
        0x0020:  0000 0000 0002 0a00 0002                 ..........
```

```
root@mininet-vm:~# tcpdump -XX -n -i h3-eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol
    decode
listening on h3-eth0, link-type EN10MB (Ethernet), capture size 262144
    bytes
```

Here we observed the behavior of the switch which is different from the hub and because of the source port mapping it sent the request to only Host 2.

We got the idea about hub code in

"/home/mininet/pox/pox/forwarding/hub.py"

on the creation of an output action that would send packets to all ports etc. and also looked at

"/home/mininet/pox/pox/forwarding/l2_learning.py"

Question 7

We submitted our filled Questionaires via ISIS.

Included Files

dump_q4.pcap