HW 3: Mininet & OpenFlow

Task 1: Defining custom topologies

What is the output of "nodes" and "net"

Output of "nodes" command:

```
mininet> nodes
available nodes are:
h1 h2 h3 h4 h5 h6 h7 h8 s1 s2 s3 s4 s5 s6 s7
```

Output of "net" command:

```
mininet> net
h1 h1-eth0:s3-eth2
h2 h2-eth0:s3-eth3
h3 h3-eth0:s4-eth2
h4 h4-eth0:s4-eth3
h5 h5-eth0:s6-eth2
h6 h6-eth0:s6-eth3
h7 h7-eth0:s7-eth2
h8 h8-eth0:s7-eth3
s1 lo:  s1-eth1:s2-eth1 s1-eth2:s5-eth1
s2 lo:  s2-eth1:s1-eth1 s2-eth2:s3-eth1 s2-eth3:s4-eth1
s3 lo:  s3-eth1:s2-eth2 s3-eth2:h1-eth0 s3-eth3:h2-eth0
s4 lo:  s4-eth1:s2-eth3 s4-eth2:h3-eth0 s4-eth3:h4-eth0
s5 lo:  s5-eth1:s1-eth2 s5-eth2:s6-eth1 s5-eth3:s7-eth1
s6 lo:  s6-eth1:s5-eth2 s6-eth2:h5-eth0 s6-eth3:h6-eth0
s7 lo:  s7-eth1:s5-eth3 s7-eth2:h7-eth0 s7-eth3:h8-eth0
```

2. What is the output of "h7 ifconfig"

```
mininet> h7 ifconfig
h7-eth0   Link encap:Ethernet  HWaddr b6:70:32:c7:93:c5
          inet addr:10.0.0.7  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::b470:32ff:fec7:93c5/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:176 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:24118 (24.1 KB)  TX bytes:648 (648.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          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:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

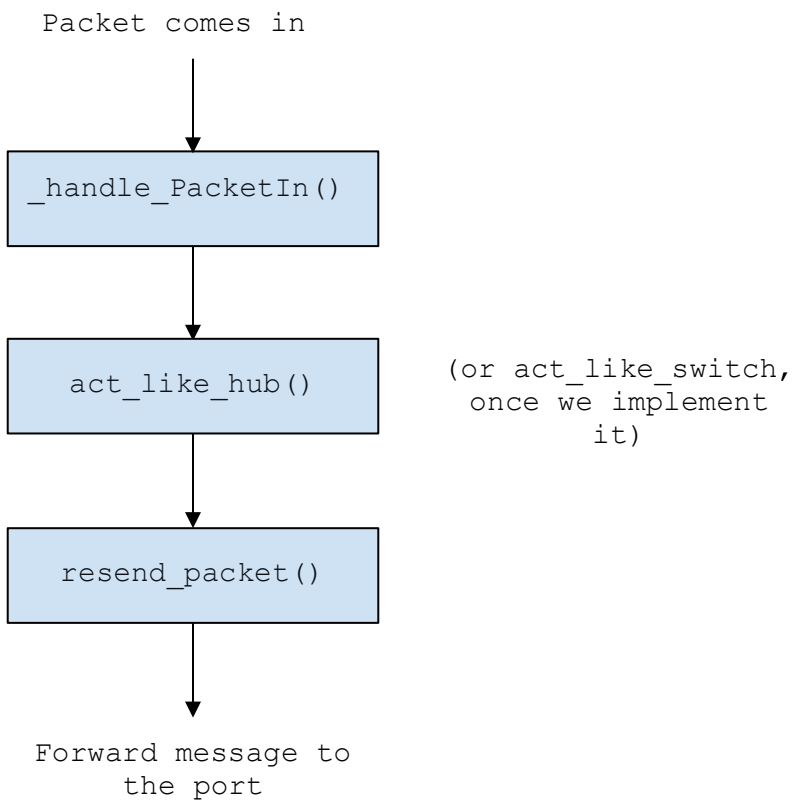
Task 2: Analyze the "of_tutorial" controller

1. Draw the function call graph of this controller. For example, once a packet comes to the controller, which function is the first to be called, which one is the second, and so forth?
We first start the POX listener using the command:

```
./pox.py log.level –DEBUG misc.of_tutorial
```

Executing this instruction initiates the 'start switch' functionality. The 'start switch' triggers the '_handle_PacketIn()' method to process incoming packets from the switch. Subsequently, the '_handle_PacketIn()' method invokes the 'act_like_hub()' method, which replicates the behavior of a hub by broadcasting packets to all ports except the input port. Following this, the 'resend_packet()' method is invoked, which modifies the packet data and performs the specified action on it. Finally, the switch is directed to transmit the modified packet to a designated port through the message.

Function call graph:
start switch : _handle_PacketIn() -> act_like_hub() -> resend_packet() -> send(msg)



2. Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).
h1 ping -c100 h2:

```
64 bytes from 10.0.0.2: icmp_seq=57 ttl=64 time=3.16 ms
64 bytes from 10.0.0.2: icmp_seq=58 ttl=64 time=1.72 ms
64 bytes from 10.0.0.2: icmp_seq=59 ttl=64 time=2.87 ms
64 bytes from 10.0.0.2: icmp_seq=60 ttl=64 time=1.34 ms
64 bytes from 10.0.0.2: icmp_seq=61 ttl=64 time=2.51 ms
64 bytes from 10.0.0.2: icmp_seq=62 ttl=64 time=3.31 ms
64 bytes from 10.0.0.2: icmp_seq=63 ttl=64 time=1.56 ms
64 bytes from 10.0.0.2: icmp_seq=64 ttl=64 time=1.12 ms
64 bytes from 10.0.0.2: icmp_seq=65 ttl=64 time=1.45 ms
64 bytes from 10.0.0.2: icmp_seq=66 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=67 ttl=64 time=2.14 ms
64 bytes from 10.0.0.2: icmp_seq=68 ttl=64 time=3.37 ms
64 bytes from 10.0.0.2: icmp_seq=69 ttl=64 time=3.19 ms
64 bytes from 10.0.0.2: icmp_seq=70 ttl=64 time=1.49 ms
64 bytes from 10.0.0.2: icmp_seq=71 ttl=64 time=1.24 ms
64 bytes from 10.0.0.2: icmp_seq=72 ttl=64 time=2.83 ms
64 bytes from 10.0.0.2: icmp_seq=73 ttl=64 time=3.18 ms
64 bytes from 10.0.0.2: icmp_seq=74 ttl=64 time=1.41 ms
64 bytes from 10.0.0.2: icmp_seq=75 ttl=64 time=2.07 ms
64 bytes from 10.0.0.2: icmp_seq=76 ttl=64 time=1.91 ms
64 bytes from 10.0.0.2: icmp_seq=77 ttl=64 time=2.79 ms
64 bytes from 10.0.0.2: icmp_seq=78 ttl=64 time=2.75 ms
64 bytes from 10.0.0.2: icmp_seq=79 ttl=64 time=3.29 ms
64 bytes from 10.0.0.2: icmp_seq=80 ttl=64 time=2.09 ms
64 bytes from 10.0.0.2: icmp_seq=81 ttl=64 time=1.57 ms
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=3.21 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=2.37 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=2.72 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=2.97 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=2.73 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=2.11 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=2.49 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=1.36 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=3.28 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=2.17 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=2.36 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=1.09 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=1.61 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=1.07 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=1.88 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=2.33 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=3.31 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=3.14 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=1.03 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 98998ms
rtt min/avg/max/mdev = 1.108/1.665/3.411/0.438 ms
mininet>
```

h1 ping -c100 h8:

```
mininet> h1 ping -c100 h8
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=7.67 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=10.48 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=4.93 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=6.83 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=6.62 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=11.27 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=7.13 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=11.39 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=10.43 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=8.03 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=7.78 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=8.71 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=6.61 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=9.38 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=6.04 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=8.29 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=5.13 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=6.53 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=8.42 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=7.04 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=5.04 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=7.07 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=10.59 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=6.89 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=5.26 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=5.93 ms
64 bytes from 10.0.0.2: icmp_seq=27 ttl=64 time=6.68 ms
64 bytes from 10.0.0.2: icmp_seq=28 ttl=64 time=5.25 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=11.03 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=8.45 ms
64 bytes from 10.0.0.2: icmp_seq=31 ttl=64 time=7.19 ms
64 bytes from 10.0.0.2: icmp_seq=32 ttl=64 time=7.76 ms
64 bytes from 10.0.0.2: icmp_seq=33 ttl=64 time=8.26 ms
64 bytes from 10.0.0.2: icmp_seq=34 ttl=64 time=8.11 ms
64 bytes from 10.0.0.2: icmp_seq=35 ttl=64 time=5.71 ms
64 bytes from 10.0.0.2: icmp_seq=36 ttl=64 time=11.68 ms
64 bytes from 10.0.0.2: icmp_seq=37 ttl=64 time=5.61 ms
64 bytes from 10.0.0.2: icmp_seq=38 ttl=64 time=11.22 ms
64 bytes from 10.0.0.2: icmp_seq=39 ttl=64 time=9.11 ms
64 bytes from 10.0.0.2: icmp_seq=40 ttl=64 time=9.98 ms
64 bytes from 10.0.0.2: icmp_seq=41 ttl=64 time=6.88 ms
64 bytes from 10.0.0.2: icmp_seq=42 ttl=64 time=7.42 ms
64 bytes from 10.0.0.2: icmp_seq=43 ttl=64 time=5.76 ms
64 bytes from 10.0.0.2: icmp_seq=44 ttl=64 time=7.37 ms
64 bytes from 10.0.0.2: icmp_seq=57 ttl=64 time=5.15 ms
64 bytes from 10.0.0.2: icmp_seq=58 ttl=64 time=5.34 ms
64 bytes from 10.0.0.2: icmp_seq=59 ttl=64 time=6.63 ms
64 bytes from 10.0.0.2: icmp_seq=60 ttl=64 time=9.31 ms
64 bytes from 10.0.0.2: icmp_seq=61 ttl=64 time=9.01 ms
64 bytes from 10.0.0.2: icmp_seq=62 ttl=64 time=11.15 ms
64 bytes from 10.0.0.2: icmp_seq=63 ttl=64 time=6.01 ms
64 bytes from 10.0.0.2: icmp_seq=64 ttl=64 time=4.55 ms
64 bytes from 10.0.0.2: icmp_seq=65 ttl=64 time=8.23 ms
64 bytes from 10.0.0.2: icmp_seq=66 ttl=64 time=10.69 ms
64 bytes from 10.0.0.2: icmp_seq=67 ttl=64 time=5.91 ms
64 bytes from 10.0.0.2: icmp_seq=68 ttl=64 time=6.98 ms
64 bytes from 10.0.0.2: icmp_seq=69 ttl=64 time=11.22 ms
64 bytes from 10.0.0.2: icmp_seq=70 ttl=64 time=10.93 ms
64 bytes from 10.0.0.2: icmp_seq=71 ttl=64 time=7.71 ms
64 bytes from 10.0.0.2: icmp_seq=72 ttl=64 time=7.93 ms
64 bytes from 10.0.0.2: icmp_seq=73 ttl=64 time=9.14 ms
64 bytes from 10.0.0.2: icmp_seq=74 ttl=64 time=10.36 ms
64 bytes from 10.0.0.2: icmp_seq=75 ttl=64 time=8.86 ms
64 bytes from 10.0.0.2: icmp_seq=76 ttl=64 time=6.58 ms
64 bytes from 10.0.0.2: icmp_seq=77 ttl=64 time=8.71 ms
64 bytes from 10.0.0.2: icmp_seq=78 ttl=64 time=11.77 ms
64 bytes from 10.0.0.2: icmp_seq=79 ttl=64 time=11.37 ms
64 bytes from 10.0.0.2: icmp_seq=80 ttl=64 time=10.74 ms
64 bytes from 10.0.0.2: icmp_seq=81 ttl=64 time=7.98 ms
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=7.32 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=8.68 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=7.69 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=5.18 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=9.25 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=8.21 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=5.82 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=7.46 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=8.17 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=7.43 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=7.81 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=6.41 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=8.92 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=11.47 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=8.91 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=6.67 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=6.78 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=10.88 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=6.45 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99855ms
rtt min/avg/max/mdev = 4.551/5.466/12.144/1.530 ms
mininet>
```

2.a How long does it take (on average) to ping for each case?

Average ping	
h1 ping h2	Average ping is 1.665 ms
h1 ping h8	Average ping is 5.466 ms

2.b What is the minimum and maximum ping you have observed?

Minimum ping	
h1 ping h2	Minimum ping observed is 1.108 ms
h1 ping h8	Minimum ping observed is 4.551 ms
Maximum ping	
h1 ping h2	Maximum ping observed is 3.411 ms
h1 ping h8	Maximum ping observed is 12.144 ms

2.c What is the difference, and why?

The duration taken for the ping operation from h1 to h8 is longer compared to the time taken for the same operation from h1 to h2. This is due to the presence of a single switch, s3, between h1 and h2, while there are multiple intermediate devices (s3, s2, s1, s5, s7) that the packets must traverse between h1 and h8.

3. Run "iperf h1 h2" and "iperf h1 h8"

Output for "iperf h1 h2" and "iperf h1 h8" commands:

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['10.7 Mbits/sec', '12.5 Mbits/sec']
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['3.15 Mbits/sec', '3.52 Mbits/sec'
mininet>
```

3.a What is "iperf" used for?

Iperf is a freely available utility that aids network administrators in assessing network performance and the quality of a network connection by measuring bandwidth. Two hosts running iperf are utilized to limit the network link, enabling the measurement of throughput between any two nodes within the network.

3.b What is the throughput for each case?

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['10.7 Mbits/sec', '12.5 Mbits/sec']
```

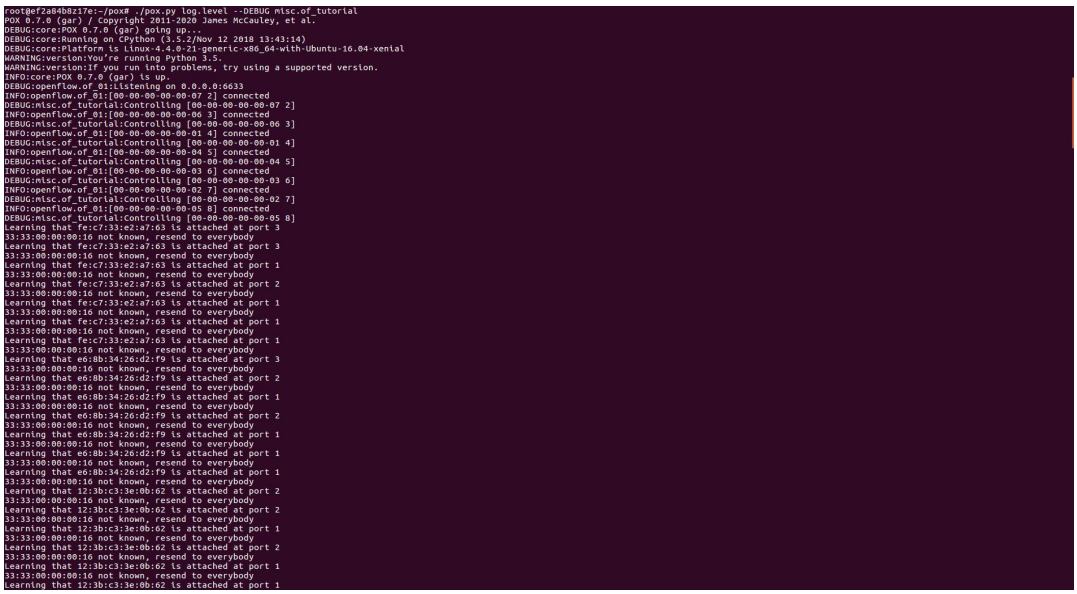
```
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['3.15 Mbits/sec', '3.52 Mbits/sec'
```

3.c What is the difference, and explain the reasons for the difference.
The reason for the higher throughput between h1 and h2, compared to h1 and h8, is due to network congestion and latency (which also causes slower ping times). The fewer hops between h1 and h2 allow for more data to be transmitted in a shorter time, while the greater number of hops between h1 and h8 results in less data being transmitted within a given time frame.

4. Which of the switches observe traffic? Please describe your way for observing such traffic on switches (e.g., adding some functions in the “of_tutorial” controller).
By adding log.info("Switch observing traffic: %s" % (self.connection) in the line number 107 "of_tutorial" controller we can see the information which helps us to observe the traffic. After viewing that, we can conclude that all the switches view the traffic, specifically when all are flooded with packets. The _handle_PacketIn function is the event listener so it's called every time a packet is received.

Task 3: MAC Learning Controller

1. Describe how the above code works, such as how the "MAC to Port" map is established. You could use a ‘ping’ example to describe the establishment process (e.g., h1 ping h2).
The act_like_switch function has the ability to "learn" or map MAC addresses, which enables the controller to efficiently direct packets to their intended destinations. When a desired MAC address is identified as the recipient, the controller maps it to a specific port for streamlined delivery. This feature enhances the controller's performance when sending packets to known addresses, as it only needs to direct the packet to the corresponding port. In instances where the destination address is unknown, the function floods the packet to all available destinations. By minimizing flooding, the MAC Learning Controller enhances ping times and throughput. To better understand the establishment process, observe the output for a single packet ping from h1 to h2.



```
33:33:00:00:00:16 not known, resend to everybody
Learning that a6:e8:e9:dd:39:24 is attached at port 1
33:33:00:00:00:16 not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
33:33:ff:0c:09:dc not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 3
33:33:00:00:00:16 not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 2
33:33:00:00:00:16 not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 1
33:33:00:00:00:16 not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 1
33:33:00:00:00:16 not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 1
33:33:00:00:00:16 not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 1
33:33:00:00:00:16 not known, resend to everybody
Learning that 1a:d1:21:83:fe:bb is attached at port 1
33:33:00:00:00:16 not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 2
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 3
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 1
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 2
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 1
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 1
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:ff:e2:a7:63 not known, resend to everybody
Learning that fa:94:b9:5b:2f:4d is attached at port 1
33:33:ff:5b:2f:4d not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
```

```
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:16 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
```

```
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
a6:e8:e9:dd:39:24 destination known. only send message to it
ff:ff:ff:ff:ff:ff not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
1a:d1:21:83:fe:bb destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
ff:ff:ff:ff:ff:ff not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
ff:ff:ff:ff:ff:ff not known, resend to everybody
1a:d1:21:83:fe:bb destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
1a:d1:21:83:fe:bb destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
1a:d1:21:83:fe:bb destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
1a:d1:21:83:fe:bb destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
1a:d1:21:83:fe:bb destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
a6:e8:e9:dd:39:24 destination known. only send message to it
33:33:00:00:00:02 not known, resend to everybody
1a:d1:21:83:fe:bb destination known. only send message to it
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
33:33:00:00:00:02 not known, resend to everybody
```

2. (Comment out all prints before doing this experiment) Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).

h1 ping -c100 h2:

```
mininet> h1 ping -c100 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=2.37 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=2.26 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=1.16 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=1.48 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=2.21 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=1.99 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=1.63 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=2.57 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=2.46 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=1.87 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=1.56 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=1.75 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=1.11 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=1.85 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=1.82 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=2.07 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=1.31 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=1.74 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=2.44 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=1.36 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=2.14 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=2.18 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=1.51 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=1.98 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=1.61 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=2.25 ms
64 bytes from 10.0.0.2: icmp_seq=27 ttl=64 time=1.53 ms
64 bytes from 10.0.0.2: icmp_seq=28 ttl=64 time=2.32 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=1.85 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=1.94 ms
64 bytes from 10.0.0.2: icmp_seq=31 ttl=64 time=1.29 ms
64 bytes from 10.0.0.2: icmp_seq=32 ttl=64 time=1.91 ms
64 bytes from 10.0.0.2: icmp_seq=33 ttl=64 time=2.22 ms
64 bytes from 10.0.0.2: icmp_seq=34 ttl=64 time=1.89 ms
64 bytes from 10.0.0.2: icmp_seq=35 ttl=64 time=2.39 ms
64 bytes from 10.0.0.2: icmp_seq=36 ttl=64 time=2.43 ms
64 bytes from 10.0.0.2: icmp_seq=37 ttl=64 time=1.22 ms
64 bytes from 10.0.0.2: icmp_seq=38 ttl=64 time=1.14 ms
64 bytes from 10.0.0.2: icmp_seq=39 ttl=64 time=2.05 ms
64 bytes from 10.0.0.2: icmp_seq=40 ttl=64 time=1.86 ms
64 bytes from 10.0.0.2: icmp_seq=41 ttl=64 time=2.24 ms
64 bytes from 10.0.0.2: icmp_seq=42 ttl=64 time=1.72 ms
64 bytes from 10.0.0.2: icmp_seq=43 ttl=64 time=1.79 ms
64 bytes from 10.0.0.2: icmp_seq=44 ttl=64 time=1.68 ms
64 bytes from 10.0.0.2: icmp_seq=45 ttl=64 time=1.26 ms
64 bytes from 10.0.0.2: icmp_seq=46 ttl=64 time=1.62 ms
64 bytes from 10.0.0.2: icmp_seq=47 ttl=64 time=1.52 ms
64 bytes from 10.0.0.2: icmp_seq=48 ttl=64 time=1.48 ms
64 bytes from 10.0.0.2: icmp_seq=49 ttl=64 time=1.21 ms
64 bytes from 10.0.0.2: icmp_seq=50 ttl=64 time=2.06 ms
64 bytes from 10.0.0.2: icmp_seq=51 ttl=64 time=1.83 ms
64 bytes from 10.0.0.2: icmp_seq=52 ttl=64 time=1.92 ms
64 bytes from 10.0.0.2: icmp_seq=53 ttl=64 time=2.76 ms
64 bytes from 10.0.0.2: icmp_seq=54 ttl=64 time=1.32 ms
64 bytes from 10.0.0.2: icmp_seq=55 ttl=64 time=1.49 ms
64 bytes from 10.0.0.2: icmp_seq=56 ttl=64 time=1.76 ms
64 bytes from 10.0.0.2: icmp_seq=57 ttl=64 time=2.68 ms
64 bytes from 10.0.0.2: icmp_seq=58 ttl=64 time=1.25 ms
64 bytes from 10.0.0.2: icmp_seq=59 ttl=64 time=2.04 ms
64 bytes from 10.0.0.2: icmp_seq=60 ttl=64 time=2.29 ms
```

```
64 bytes from 10.0.0.2: icmp_seq=57 ttl=64 time=2.08 ms
64 bytes from 10.0.0.2: icmp_seq=58 ttl=64 time=1.25 ms
64 bytes from 10.0.0.2: icmp_seq=59 ttl=64 time=2.04 ms
64 bytes from 10.0.0.2: icmp_seq=60 ttl=64 time=2.29 ms
64 bytes from 10.0.0.2: icmp_seq=61 ttl=64 time=1.57 ms
64 bytes from 10.0.0.2: icmp_seq=62 ttl=64 time=1.58 ms
64 bytes from 10.0.0.2: icmp_seq=63 ttl=64 time=1.38 ms
64 bytes from 10.0.0.2: icmp_seq=64 ttl=64 time=1.37 ms
64 bytes from 10.0.0.2: icmp_seq=65 ttl=64 time=2.33 ms
64 bytes from 10.0.0.2: icmp_seq=66 ttl=64 time=2.49 ms
64 bytes from 10.0.0.2: icmp_seq=67 ttl=64 time=2.28 ms
64 bytes from 10.0.0.2: icmp_seq=68 ttl=64 time=2.11 ms
64 bytes from 10.0.0.2: icmp_seq=69 ttl=64 time=2.36 ms
64 bytes from 10.0.0.2: icmp_seq=70 ttl=64 time=1.55 ms
64 bytes from 10.0.0.2: icmp_seq=71 ttl=64 time=1.45 ms
64 bytes from 10.0.0.2: icmp_seq=72 ttl=64 time=1.59 ms
64 bytes from 10.0.0.2: icmp_seq=73 ttl=64 time=1.71 ms
64 bytes from 10.0.0.2: icmp_seq=74 ttl=64 time=2.03 ms
64 bytes from 10.0.0.2: icmp_seq=75 ttl=64 time=2.13 ms
64 bytes from 10.0.0.2: icmp_seq=76 ttl=64 time=1.42 ms
64 bytes from 10.0.0.2: icmp_seq=77 ttl=64 time=1.97 ms
64 bytes from 10.0.0.2: icmp_seq=78 ttl=64 time=2.17 ms
64 bytes from 10.0.0.2: icmp_seq=79 ttl=64 time=2.27 ms
64 bytes from 10.0.0.2: icmp_seq=80 ttl=64 time=1.18 ms
64 bytes from 10.0.0.2: icmp_seq=81 ttl=64 time=2.47 ms
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=2.37 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=1.28 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=2.15 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=1.26 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=1.54 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=1.81 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=1.15 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=1.73 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=1.84 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=2.17 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=1.36 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=1.65 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=2.23 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=2.46 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=1.95 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=2.19 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=1.71 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=1.47 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=1.93 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99158ms
rtt min/avg/max/mdev = 1.118/1.822/2.541/0.265 ms
mininet>
```

h1 ping -c100 h8:

```
mininet> h1 ping -c100 h8
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=9.15 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=6.34 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=12.03 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=5.61 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=9.36 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=4.69 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=6.81 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=6.66 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=9.82 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=7.83 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=4.68 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=4.05 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=6.36 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=6.39 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=6.07 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=8.07 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=4.38 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=5.57 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=8.73 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=11.81 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=5.87 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=12.25 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=5.19 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=6.23 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=11.44 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=10.14 ms
64 bytes from 10.0.0.2: icmp_seq=27 ttl=64 time=9.24 ms
64 bytes from 10.0.0.2: icmp_seq=28 ttl=64 time=12.81 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=4.83 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=8.58 ms
64 bytes from 10.0.0.2: icmp_seq=31 ttl=64 time=8.66 ms
64 bytes from 10.0.0.2: icmp_seq=32 ttl=64 time=10.12 ms
64 bytes from 10.0.0.2: icmp_seq=33 ttl=64 time=3.58 ms
64 bytes from 10.0.0.2: icmp_seq=34 ttl=64 time=4.35 ms
64 bytes from 10.0.0.2: icmp_seq=35 ttl=64 time=12.89 ms
64 bytes from 10.0.0.2: icmp_seq=36 ttl=64 time=8.01 ms
64 bytes from 10.0.0.2: icmp_seq=37 ttl=64 time=4.77 ms
64 bytes from 10.0.0.2: icmp_seq=38 ttl=64 time=5.81 ms
64 bytes from 10.0.0.2: icmp_seq=39 ttl=64 time=7.97 ms
64 bytes from 10.0.0.2: icmp_seq=40 ttl=64 time=8.45 ms
64 bytes from 10.0.0.2: icmp_seq=41 ttl=64 time=12.55 ms
64 bytes from 10.0.0.2: icmp_seq=42 ttl=64 time=7.44 ms
64 bytes from 10.0.0.2: icmp_seq=43 ttl=64 time=6.27 ms
64 bytes from 10.0.0.2: icmp_seq=44 ttl=64 time=4.23 ms
64 bytes from 10.0.0.2: icmp_seq=45 ttl=64 time=9.02 ms
64 bytes from 10.0.0.2: icmp_seq=46 ttl=64 time=6.97 ms
64 bytes from 10.0.0.2: icmp_seq=47 ttl=64 time=11.46 ms
64 bytes from 10.0.0.2: icmp_seq=48 ttl=64 time=3.61 ms
64 bytes from 10.0.0.2: icmp_seq=49 ttl=64 time=4.64 ms
64 bytes from 10.0.0.2: icmp_seq=50 ttl=64 time=5.71 ms
64 bytes from 10.0.0.2: icmp_seq=51 ttl=64 time=10.76 ms
64 bytes from 10.0.0.2: icmp_seq=52 ttl=64 time=11.96 ms
64 bytes from 10.0.0.2: icmp_seq=53 ttl=64 time=4.59 ms
64 bytes from 10.0.0.2: icmp_seq=54 ttl=64 time=7.71 ms
64 bytes from 10.0.0.2: icmp_seq=55 ttl=64 time=11.3 ms
64 bytes from 10.0.0.2: icmp_seq=56 ttl=64 time=8.28 ms
64 bytes from 10.0.0.2: icmp_seq=57 ttl=64 time=7.09 ms
64 bytes from 10.0.0.2: icmp_seq=58 ttl=64 time=4.52 ms
64 bytes from 10.0.0.2: icmp_seq=59 ttl=64 time=4.97 ms
64 bytes from 10.0.0.2: icmp_seq=60 ttl=64 time=5.26 ms
```

2.a How long did it take (on average) to ping for each case?

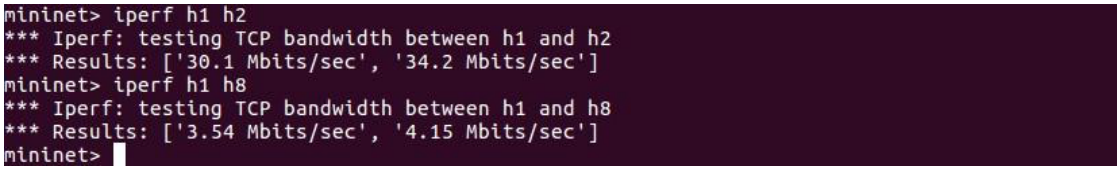
Average ping	
h1 ping h2	Average ping is 1.822 ms
h1 ping h8	Average ping is 4.275 ms

2.b What is the minimum and maximum ping you have observed?

Minimum ping	
h1 ping h2	Minimum ping observed is 1.118 ms
h1 ping h8	Minimum ping observed is 3.531 ms
Maximum ping	
h1 ping h2	Maximum ping observed 2.541 is ms
h1 ping h8	Maximum ping observed 12.893 is ms

2.c Any difference from Task 2 and why do you think there is a change if there is?
In comparison to task 2, the time taken for h1 to ping h2 is slightly shorter in task 3, though the difference is not substantial. However, in the case of h1 and h8, there is a noticeable difference in the ping time values due to the greater number of switches involved. The lower ping times in task 3 can be attributed to the fact that only the initial few packets are flooded. Once the destination MAC address is identified and mapped, switches only resend packets to the corresponding port designated in the "mac_to_port" map. Consequently, subsequent pings experience less network congestion and faster delivery times.

3. Run "iperf h1 h2" and "iperf h1 h8".



3.a What is the throughput for each case?

mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['30.1 Mbits/sec', '34.2 Mbits/sec']
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['3.54 Mbits/sec', '4.15 Mbits/sec'

3.b What is the difference from Task 2 and why do you think there is a change if there is?
In both scenarios, task 3 demonstrates a higher throughput than task 2. This is primarily due to the reduced network congestion in task 3, since flooding of packets is minimized after the mac_to_port map has learned all the relevant ports. As a result, the switches are not as heavily burdened. For h1 and h2, we can observe an average improvement in throughputs of approximately three times in tasks 1 and 2, due to more pre-computed and learned routes with changes in the controller. However, in the case of h1 and h8, the improvement is not significant, but there is a slight improvement, likely due to the number of hops and packet loss.