COEN 241 HW 3
Mininet & OpenFlow
Justin Li

Task 1:

1. nodes & net:

```
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 h5 h6 h7 h8 s1 s2 s3 s4 s5 s6 s7
mininet> net
h1 h1-eth0:s3-eth1
h2 h2-eth0:s3-eth2
h3 h3-eth0:s4-eth1
h4 h4-eth0:s4-eth2
h5 h5-eth0:s6-eth1
h6 h6-eth0:s6-eth2
h7 h7-eth0:s7-eth1
h8 h8-eth0:s7-eth2
s1 lo:  s1-eth1:s2-eth3 s1-eth2:s5-eth3
s2 lo:  s2-eth1:s3-eth3 s2-eth2:s4-eth3 s2-eth3:s1-eth1
s3 lo:  s3-eth1:h1-eth0 s3-eth2:h2-eth0 s3-eth3:s2-eth1
s4 lo:  s4-eth1:h3-eth0 s4-eth2:h4-eth0 s4-eth3:s2-eth2
s5 lo:  s5-eth1:s6-eth3 s5-eth2:s7-eth3 s5-eth3:s1-eth2
s6 lo:  s6-eth1:h5-eth0 s6-eth2:h6-eth0 s6-eth3:s5-eth1
s7 lo:  s7-eth1:h7-eth0 s7-eth2:h8-eth0 s7-eth3:s5-eth2
c0
mininet>
```

2. h7 ifconfig:

```
mininet> h7 ifconfig
h7-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.7  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::8042:e9ff:fef2:1f55  prefixlen 64  scopeid 0x20<link>
        ether 82:42:e9:f2:1f:55  txqueuelen 1000  (Ethernet)
        RX packets 75  bytes 5722 (5.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

mininet> _
```

Task 2:

1. _handle_PacketIn() → act_like_hub() → resend_packet()
2. Ping

```
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99204ms
rtt min/avg/max/mdev = 1.117/1.559/4.912/0.468 ms
mininet>
```

```
--- 10.0.0.8 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99182ms
rtt min/avg/max/mdev = 5.608/7.061/14.275/1.701 ms
mininet> _
```

   a. h1 ping h2 avg: 1.559 ms
      h1 ping h8 avg: 7.061 ms
   b. h1 ping h2 min: 1.117 ms
      h1 ping h2 max: 4.912 ms
      h1 ping h8 min: 5.608 ms
      h1 ping h8 max: 14.275 ms
   c. Difference between max and min: 13.158 ms
      Explanation: h1 and h8 are much further away from each other in the topology compared to h1 and h2, as h1 and h8 are 6 connections away from each other while h1 and h2 are only 2 connections away.

3. Iperf

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
.*** Results: ['6.95 Mbits/sec', '8.22 Mbits/sec']
mininet>
```

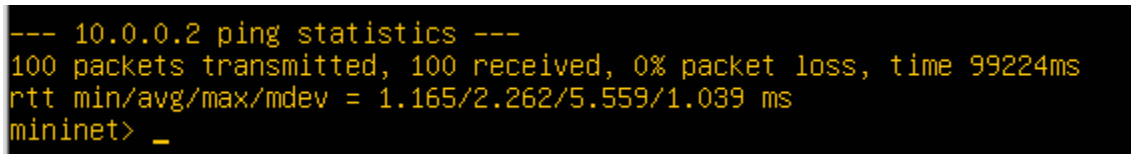
```
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['2.40 Mbits/sec', '2.89 Mbits/sec']
mininet> _
```

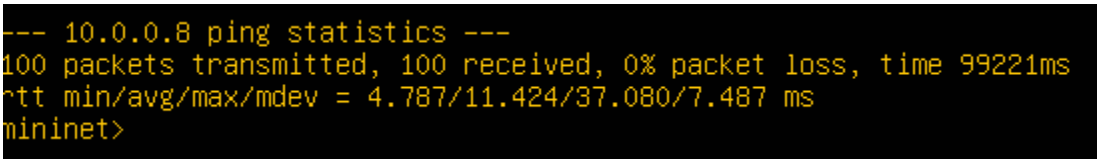
   a. iperf is used to test the TCP bandwidth between hosts.
   b. iperf h1 h2: [6.95, 8.22] Mbits/sec
      iperf h1 h8: [2.4, 2.89] Mbits/sec
   c. The difference is 4.55 for the server send speed and 5.33 for the client send speed. Again, the difference can be attributed to the fact that h1 and h8 are further away from each other in the topology compared to h1 and h2, causing their bandwidth to be slower.

4. All of the switches observe traffic to some extent. Each switch is connected to other switches or hosts using virtual ethernet cables, where the traffic occurs. In of_tutorial.py, there is a function called “act_like_switch()” which can be modified to implement switch-like behavior.

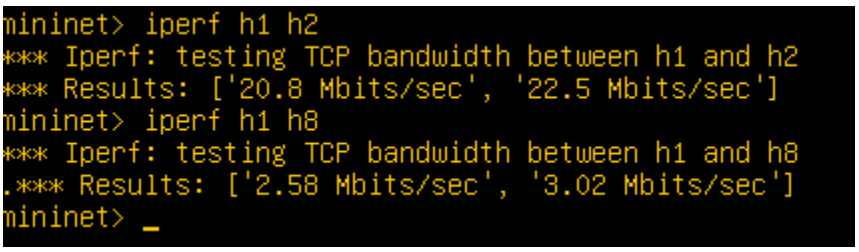
Task 3:

1. First, we check to see if packet.src, or the source MAC, is already stored in the self.mac_to_port dictionary instance variable or not. If the key doesn't exist yet, then the program uses the source MAC as a new key which is then associated with the port of a specific switch given by the .in_port value of the packet_in parameter, which is stored as the value.
2. Ping





   a. h1 ping h2 avg: 2.262 ms
      h1 ping h8 avg: 11.424 ms
   b. h1 ping h2 min: 1.165 ms
      h1 ping h2 max: 5.559 ms
      h1 ping h8 min: 4.787 ms
      h1 ping h8 max: 37.08 ms
   c. This time, the difference between max and min ping values is ~36ms, which is relatively a lot higher than the previous 14ms difference. This makes sense because we are taking extra steps in associating the new MAC addresses with different ports, thus requiring a longer time.
3. Iperf



   a. iperf h1 h2: [20.8, 22.5] Mbits/sec
      iperf h1 h8: [2.58, 3.02] Mbits/sec
   b. The bandwidth between hosts h1 and h2 are a lot higher in comparison to before. This makes sense, as once the port number is learned, we no longer have to flood the network with packets, as before we were sending them through every outgoing link, which took a longer amount of time. Thus, it is now a lot quicker to send the packets to the right destination.