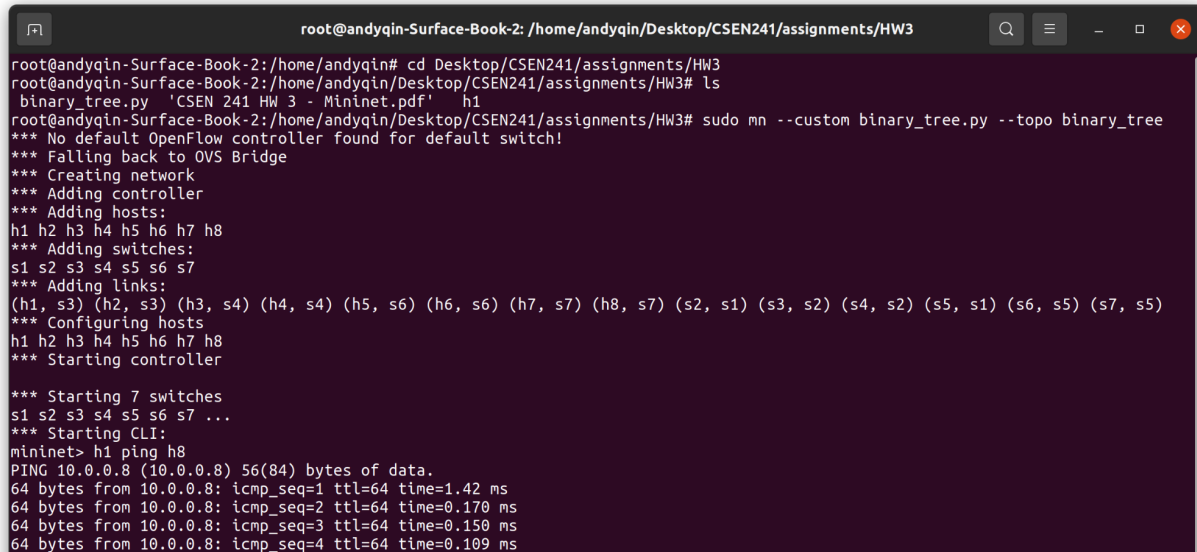


# CSEN 241 – Cloud Computing

## Homework #3

### Task 1: Defining custom topologies

\$ sudo mn --custom binary\_tree.py --topo binary\_tree



```

root@andyqin-Surface-Book-2: /home/andyqin/Desktop/CSEN241/assignments/HW3
root@andyqin-Surface-Book-2:/home/andyqin/Desktop/CSEN241/assignments/HW3# ls
binary_tree.py  'CSEN 241 HW 3 - Mininet.pdf'  h1
root@andyqin-Surface-Book-2:/home/andyqin/Desktop/CSEN241/assignments/HW3# sudo mn --custom binary_tree.py --topo binary_tree
*** No default OpenFlow controller found for default switch!
*** Falling back to OVS Bridge
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(h1, s3) (h2, s3) (h3, s4) (h4, s4) (h5, s6) (h6, s6) (h7, s7) (h8, s7) (s2, s1) (s3, s2) (s4, s2) (s5, s1) (s6, s5) (s7, s5)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller

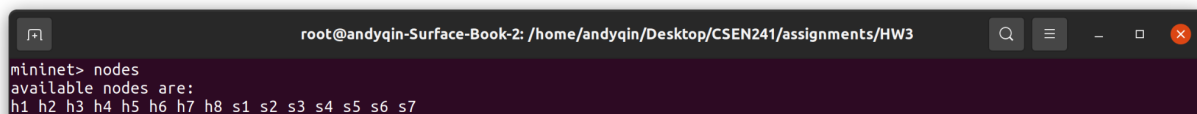
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
mininet> h1 ping h8
PING 10.0.0.8 (10.0.0.8) 56(84) bytes of data.
64 bytes from 10.0.0.8: icmp_seq=1 ttl=64 time=1.42 ms
64 bytes from 10.0.0.8: icmp_seq=2 ttl=64 time=0.170 ms
64 bytes from 10.0.0.8: icmp_seq=3 ttl=64 time=0.150 ms
64 bytes from 10.0.0.8: icmp_seq=4 ttl=64 time=0.109 ms

```

### Questions

1. What is the output of “nodes” and “net”

\$ node



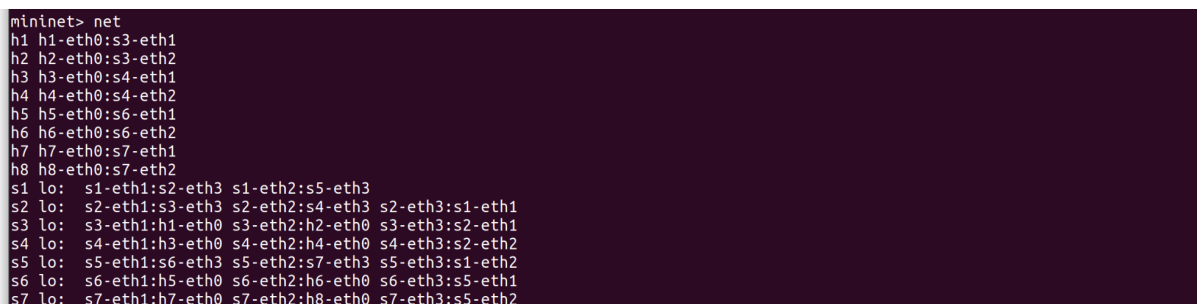
```

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

```

Lists all devices in the network topology including switches hosts, and controller

\$ net



```

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

```

Lists all the links created within the network topology

## 2. What is the output of “h7 ifconfig”

\$ 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::dca4:51ff:fe63:cc99 prefixlen 64 scopeid 0x20<link>
    ether de:a4:51:63:cc:99 txqueuelen 1000 (Ethernet)
    RX packets 258 bytes 35221 (35.2 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

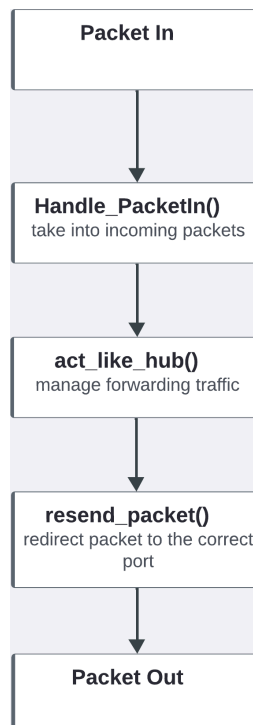
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped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

Lists available network configurations

## Task 2: Install POX

### Questions

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?



Answer: When a packet arrives, it is first handled by **Handle\_PacketIn()**, then **act\_like\_hub()**, then **resend\_packet()**

2. Have h1 ping h2, and h1 ping h8 for 100 times (e.g., h1 ping -c100 p2).
  - a. How long does it take (on average) to ping for each case?
  - b. What is the minimum and maximum ping you have observed?
  - c. What is the difference, and why?

\$h1 ping -c 100 h2

```
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99738ms
rtt min/avg/max/mdev = 3.886/4.936/6.623/0.436 ms
mininet>
```

\$h1 ping -c 100 h8

```
--- 10.0.0.8 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99219ms
rtt min/avg/max/mdev = 9.975/18.909/22.953/2.825 ms
mininet>
```

- a) Average time to ping
  - i) h1 to h2: **4.936 ms**
  - ii) h1 to h8: **18.909 ms**
- b) Min / Max time to ping
  - i) Min
    - h1 to h2: **3.886 ms**
    - h1 to h8: **9.9975 ms**
  - ii) Max
    - h1 to h2: **6.623 ms**
    - h1 to h8: **22.953 ms**
- c) Difference
  - i) h1 to h2 has much faster TurnAround Time as compared to h1 to h8
  - ii) This is probably due to the amount of switching and intermediate routes the packet has to travel before arriving at the destination
    - 1) h1 to h2: h1 → s3 → h2
    - 2) h1 to h8: h1 → s3 → s2 → s1 → s5 → s7 → h8
  - iii) While there is only 1 switch between h1 and h2, there are 5 switches between h1 and h8, causing more traveling and latency.
  - iv) The amount of difference between ping time is on average around **14 ms**

3. Run “iperf h1 h2” and “iperf h1 h8”
  - a. What is “iperf” used for?
  - b. What is the throughput for each case?
  - c. What is the difference, and explain the reasons for the difference.
- a) Iperf is a commonly used network testing tool that can create TCP and UDP data streams and measure the throughput of a network that is carrying them for performance evaluation
- b) Throughput for each case
  - i) \$ iperf h1 h2

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['13.9 Mbits/sec', '13.7 Mbits/sec']
```

- ii) \$ iperf h1 h8

```
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['3.39 Mbits/sec', '3.29 Mbits/sec']
```

- c) Difference between
    - i) h1 to h2 has higher bandwidth and throughput than h1 to h8
    - ii) This is reasonable because there are more intermediate switches between h1 to h8 which increases the chance of getting traffic and congestion
    - iii) With delay in each switch, packets might have to be queued somewhere in the middle to wait for next send instruction
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).

**Answer:** most traffic comes from s1, s2, s3, s5, and s7, as this information can be observed from the handle\_PacketIn() function where all packet traffic come in.

## Task 3: MAC Learning Controller

### Questions

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).
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).
  - a. How long did it take (on average) to ping for each case?
  - b. What is the minimum and maximum ping you have observed?

- c. Any difference from Task 2 and why do you think there is a change if there is?
3. Q.3 Run “iperf h1 h2” and “iperf h1 h8”.
  - a. What is the throughput for each case?
  - b. What is the difference from Task 2 and why do you think there is a change if there is?

1. Every time a ping happens, a packet is sent from source to destination; instead of just passing the packets, the network also tracks a map of the routes and MAC addresses of hosts from source to destination so that if later packets have the same routes, the packet can be sent directly to the right place with a lookup in the **mac\_to\_port** dictionary instead of having to be flooded

2. Pings

\$h1 ping -c 100 h2

```
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=5.53 ms
--- 10.0.0.2 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99281ms
rtt min/avg/max/mdev = 2.610/4.852/5.652/0.510 ms
mininet> 
```

\$h1 ping -c 100 h8

```
--- 10.0.0.8 ping statistics ---
100 packets transmitted, 100 received, 0% packet loss, time 99136ms
rtt min/avg/max/mdev = 8.915/18.374/20.694/1.345 ms
mininet> iperf h1 h2
```

- a) Average time to ping
  - i) h1 to h2: **4.852 ms**
  - ii) h1 to h8: **18.374 ms**
- b) Min / Max Time to Ping
  - i) Min
    - h1 to h2: **2.610 ms**
    - h1 to h8: **8.915 ms**
  - ii) Max
    - h1 to h2: **5.652 ms**
    - h1 to h8: **20.694 ms**
- c) Difference Between
  - i) In general, task completion time is faster in task 3 as compared to task 2

- ii) With the new packet controlling method, we can see a great decrease in average ping time (1s diff) from h1 to h2 and h1 to h8 and a slight decrease in min and max ping time.
- iii) This is because of the memorization of port mapping on the controller that allows less flooding and more direct forwarding from hosts to hosts.

### 3. Iperf

#### a. Throughput for each case

##### i. \$ iperf h1 h2

```
mininet> iperf h1 h2
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['41.8 Mbits/sec', '41.4 Mbits/sec']
```

##### ii. \$ iperf h1 h8

```
mininet> iperf h1 h8
*** Iperf: testing TCP bandwidth between h1 and h8
*** Results: ['4.50 Mbits/sec', '4.33 Mbits/sec']
```

#### b. Difference between

- i. h1 to h2 has higher bandwidth and throughput than h1 to h8
- ii. This is reasonable because there are more intermediate switches between h1 to h8 which increases the chance of getting traffic and congestion
- iii. With a delay in each switch, packets might have to be queued somewhere in the middle to wait for the next send instruction