Mokhtari (9831143) – Computer Networks 2 – PRJ 01

P1:
The Mininet was successfully installed, exactly as TA described in the tutorial video!

P2:
The output of commands entered in the Linux terminal is displayed as follows:

```
melmo@melmo:~$ sudo mn --topo single,3
*** Creating network
*** Adding controller
*** 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> nodes
available nodes are:
c0 h1 h2 h3 s1
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s1-eth3
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:h3-eth0
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=5975>
<Host h2: h2-eth0:10.0.0.2 pid=5977>
<Host h3: h3-eth0:10.0.0.3 pid=5979>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-
eth3:None pid=5984>
<Controller c0: 127.0.0.1:6653 pid=5968>
```

(1) **mininet> nodes**:
As shown in the output above, this command displays the nodes available in the current Mininet network. When we use this command, we get the following results: c0, h1, h2, h3, and s1. These nodes are available in our three-host single topology.

(2) **mininet> net**:
This command displays a list of all available network connections and links. That is, it depicts how network elements in the simulated network are linked to one another. In our case, it specifically demonstrates:

1. Host h1 is connected using its network interface h1-eth0 to the switch on interface s1-eth1.
2. Host h2 is connected using its network interface h2-eth0 to the switch on interface s1-eth2.
3. Host h3 is connected using its network interface h3-eth0 to the switch on interface s1-eth3.
4. Switch s1:
   a) has a loopback interface lo.
   b) connects to h1-eth0 through interface s1-eth1.
   c) connects to h2-eth0 through interface s1-eth2.
   d) connects to h3-eth0 through interface s1-eth3.
5. Controller c0 is the network's brain, and it knows everything about it. A controller tells network switches how to forward or drop packets.

(3) **mininet> dump**:
This command displays the dump data for all nodes in the current Mininet network. In other words, it contains information about the simulated network's nodes, switches, and controllers. In our case, it specifically demonstrates:

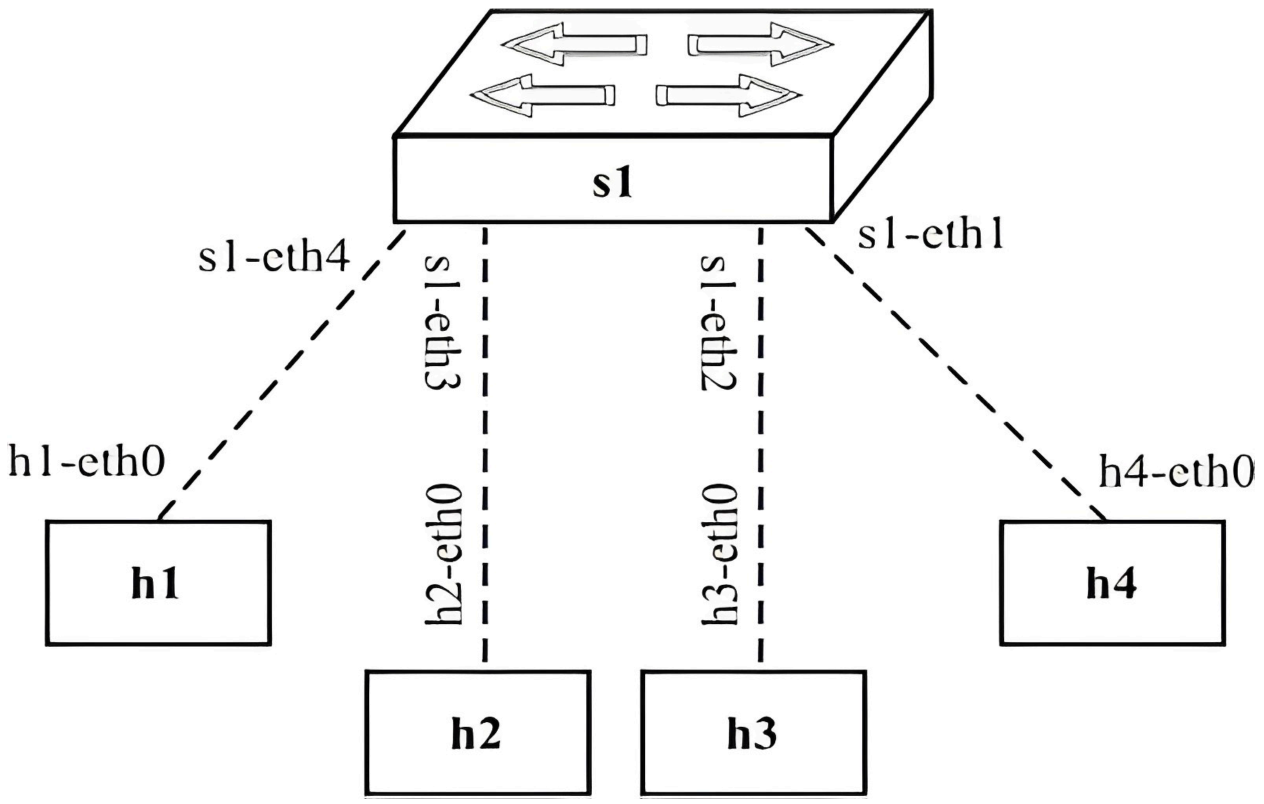
1. Host h1 has the IP address 10.0.0.1 and a pid number of 5975 through its network interface h1-eth0.
2. Host h2 has the IP address 10.0.0.2 and a pid number of 5977 through its network interface h2-eth0.
3. Host h3 has the IP address 10.0.0.3 and a pid number of 5979 through its network interface h3-eth0.
4. Switch s1 has a loopback interface with the IP address of 127.0.0.1, a network interface of s1-eth1 without any IP address, a network interface of s1-eth2 without any IP address, a network interface of s1-eth3 without any IP address, and a PID of 5984.
5. Controller c0 has the IP address 127.0.0.1 and a port number of 6653 with a PID of 5968.

P3:
Mininet topologies include minimal, single, reversed, linear, tree, and others. Let's go through the last three one by one:

I. **Reversed Topology:**
It is similar to a single connection, but the order of connections between hosts and switches is reversed.
To run a reversed topology, we use the command in the same way as a single one, but with a few small changes. The output is depicted below:

```
melmo@melmo:~$ sudo mn --topo reversed,4
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 s1
mininet> net
h1 h1-eth0:s1-eth4
h2 h2-eth0:s1-eth3
h3 h3-eth0:s1-eth2
h4 h4-eth0:s1-eth1
s1 lo:  s1-eth1:h4-eth0 s1-eth2:h3-eth0 s1-eth3:h2-eth0 s1-
eth4:h1-eth0
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=6794>
<Host h2: h2-eth0:10.0.0.2 pid=6796>
<Host h3: h3-eth0:10.0.0.3 pid=6798>
<Host h4: h4-eth0:10.0.0.4 pid=6800>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-
eth3:None,s1-eth4:None pid=6805>
<Controller c0: 127.0.0.1:6653 pid=6787>
```

The reversed topology network schematic of the previous output code is depicted in the figure below:

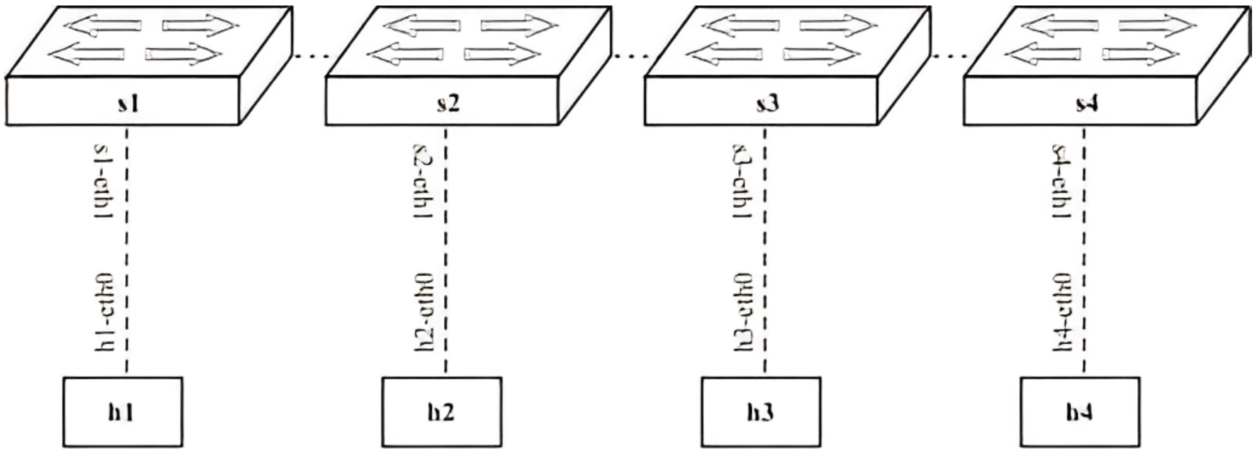


II. Linear Topology:

It is made up of N switches and N hosts. It also establishes a connection between each switch and each host, as well as between the switches.
The commands we use to run a linear topology are listed below, along with the outcomes:

```
melmo@melmo:~$ sudo mn --topo linear,4
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (s2, s1) (s3, s2) (s4, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 s1 s2 s3 s4
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
h4 h4-eth0:s4-eth1
s1 lo:  s1-eth1:h1-eth0 s1-eth2:s2-eth2
s2 lo:  s2-eth1:h2-eth0 s2-eth2:s1-eth2 s2-eth3:s3-eth2
s3 lo:  s3-eth1:h3-eth0 s3-eth2:s2-eth3 s3-eth3:s4-eth2
s4 lo:  s4-eth1:h4-eth0 s4-eth2:s3-eth3
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=7785>
<Host h2: h2-eth0:10.0.0.2 pid=7787>
<Host h3: h3-eth0:10.0.0.3 pid=7789>
<Host h4: h4-eth0:10.0.0.4 pid=7791>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=7796>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-
eth3:None pid=7799>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-
eth3:None pid=7802>
<OVSSwitch s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None pid=7805>
<Controller c0: 127.0.0.1:6653 pid=7777>
```

The linear topology network schematic of the previous output code is depicted in the figure below:

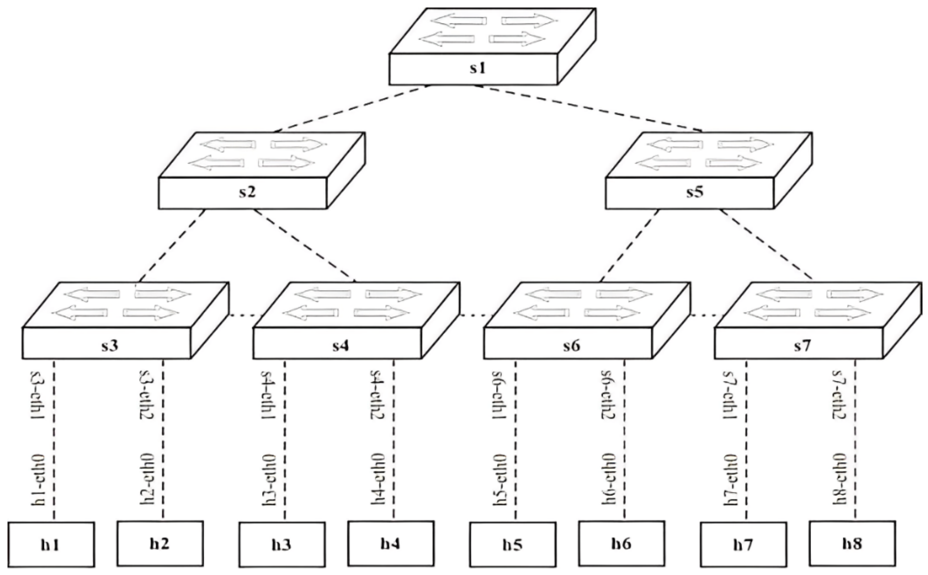


I. **Tree Topology:**
The tree topology is a multilevel topology with N levels of switches, with hosts connected to lower-level switches (two hosts are attached to each switch).
The commands listed below can be used to create tree topology:

```
melmo@melmo:~$ sudo mn --topo tree,3
*** 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:
(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3)
(s4, h4) (s5, s6) (s5, s7) (s6, h5) (s6, h6) (s7, h7) (s7, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
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> dump
<Host h1: h1-eth0:10.0.0.1 pid=8970>
<Host h2: h2-eth0:10.0.0.2 pid=8972>
<Host h3: h3-eth0:10.0.0.3 pid=8974>
<Host h4: h4-eth0:10.0.0.4 pid=8976>
```

```
<Host h5: h5-eth0:10.0.0.5 pid=8978>
<Host h6: h6-eth0:10.0.0.6 pid=8980>
<Host h7: h7-eth0:10.0.0.7 pid=8982>
<Host h8: h8-eth0:10.0.0.8 pid=8984>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=8989>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-
eth3:None pid=8992>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-
eth3:None pid=8995>
<OVSSwitch s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None,s4-
eth3:None pid=8998>
<OVSSwitch s5: lo:127.0.0.1,s5-eth1:None,s5-eth2:None,s5-
eth3:None pid=9001>
<OVSSwitch s6: lo:127.0.0.1,s6-eth1:None,s6-eth2:None,s6-
eth3:None pid=9004>
<OVSSwitch s7: lo:127.0.0.1,s7-eth1:None,s7-eth2:None,s7-
eth3:None pid=9007>
<Controller c0: 127.0.0.1:6653 pid=8963>
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8
h2 -> h1 h3 h4 h5 h6 h7 h8
h3 -> h1 h2 h4 h5 h6 h7 h8
h4 -> h1 h2 h3 h5 h6 h7 h8
h5 -> h1 h2 h3 h4 h6 h7 h8
h6 -> h1 h2 h3 h4 h5 h7 h8
h7 -> h1 h2 h3 h4 h5 h6 h8
h8 -> h1 h2 h3 h4 h5 h6 h7
*** Results: 0% dropped (56/56 received)
```

The tree topology network schematic of the previous output code is depicted below:


s1
s2
s5
s3
s4
s6
s7
s3-eth1
s3-eth2
s4-eth1
s4-eth2
s6-eth1
s6-eth2
s7-eth1
s7-eth2
h1-eth0
h2-eth0
h3-eth0
h4-eth0
h5-eth0
h6-eth0
h7-eth0
h8-eth0
h1
h2
h3
h4
h5
h6
h7
h8