Practical 7- Defining data, control and application planes in software defined networking in Openflow.

→ Mininet is a network emulation software that allows you to launch a virtual network with switches, hosts and an SDN controller all with single command.

→ In order to start a minimal topology, following command is used:

- `sudo mn`

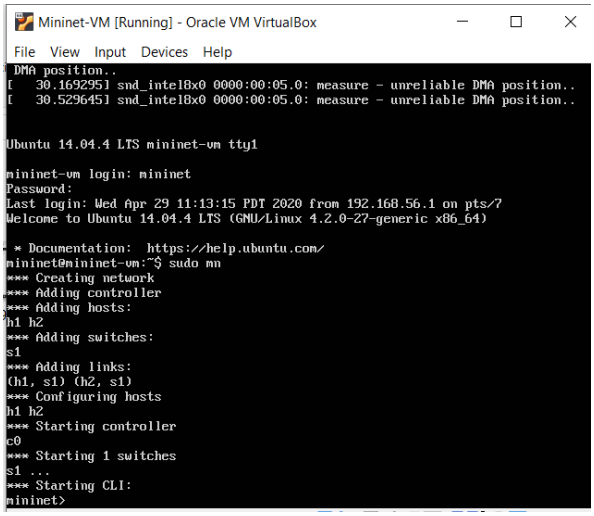

Figure 1: A minimal topology

→ Here, mn is a launch script that executes python.

→ The topology in figure 1 consists of two hosts, one switch and one central controller. The setup is shown in figure 2.

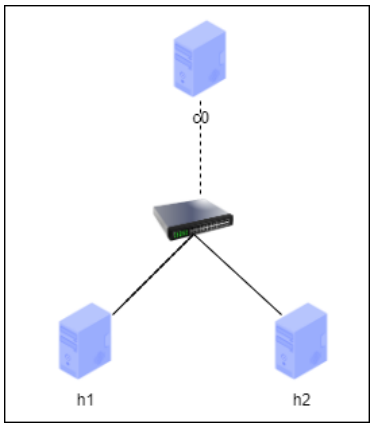

Figure 2

→ We can check out all the nodes available by simply typing the “nodes” command.

- mininet> nodes

→ We can check out the links in the network with the help of command:

- mininet> net

→ We can display all the information about the nodes with the help of:

- mininet> dump

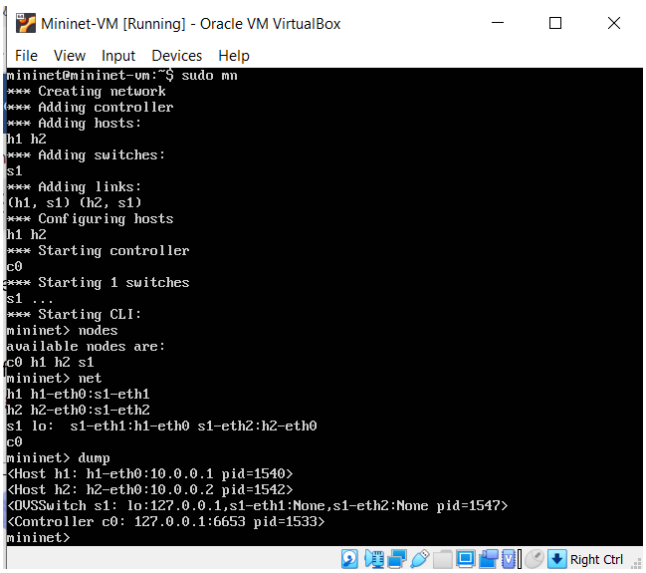

Figure 3

→ We can check the ip addresses of nodes and print out process list from a host using following commands:

- mininet> h1 ifconfig -a
- mininet> h1 ps –a

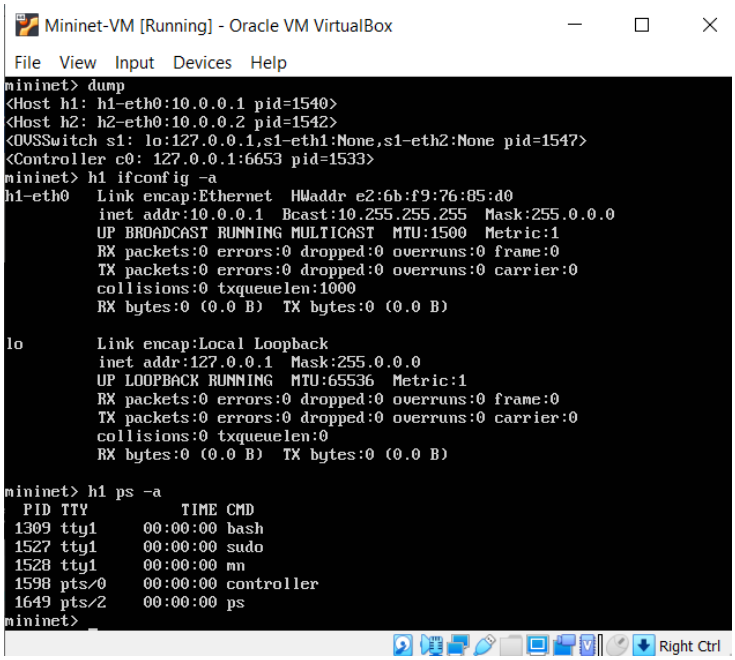

Figure 4: Host ip address and process list

→ Now we test connectivity between hosts by writing ping command as shown in the figure 5.

→ We can observe in figure 5 that, during the first time the hosts are pinged, the time taken is 2.38ms. This is because the openflow switch sends it first packet's header and forwards to the controller for actions to perform. The controller responds with appropriate actions like in this case forwarding packet to host 2. This entry then remains cached in the switch for certain time. So, the next time packet is sent from h1 to h2 it goes directly to its destination without going to the controller.

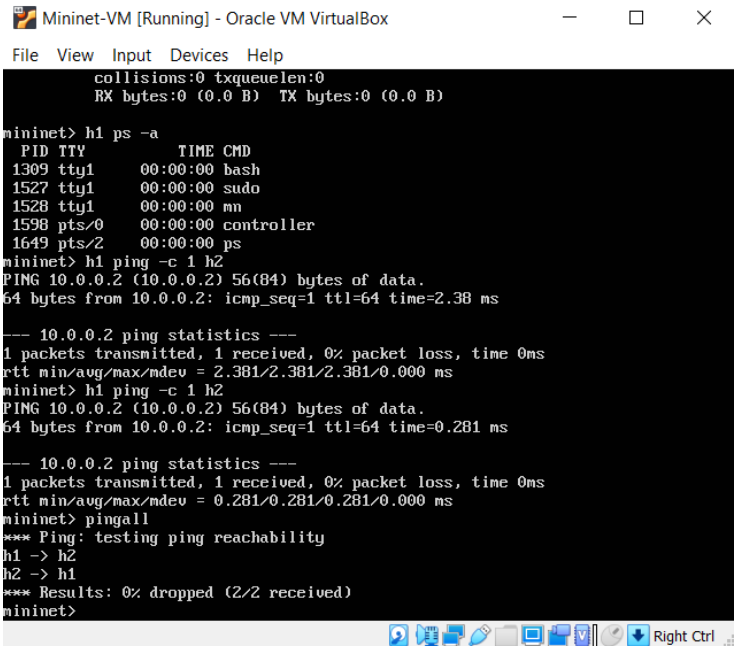

Figure 5: Testing connectivity between hosts

→ We can configure links between our hosts like changing bandwidth and introducing delay as follows:

- sudo mn -link tc,bw=[bandwidth],delay=[delay_in_millisecond]

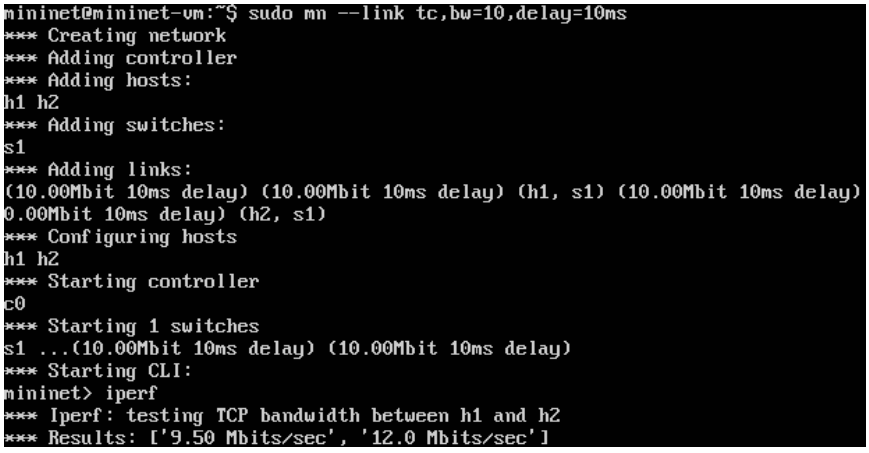

Figure 6: Configure link

→ With iperf command we can see tcp bandwidth between hosts as shown in figure 6.

→ Now we create our own custom topology as shown in figure 7.

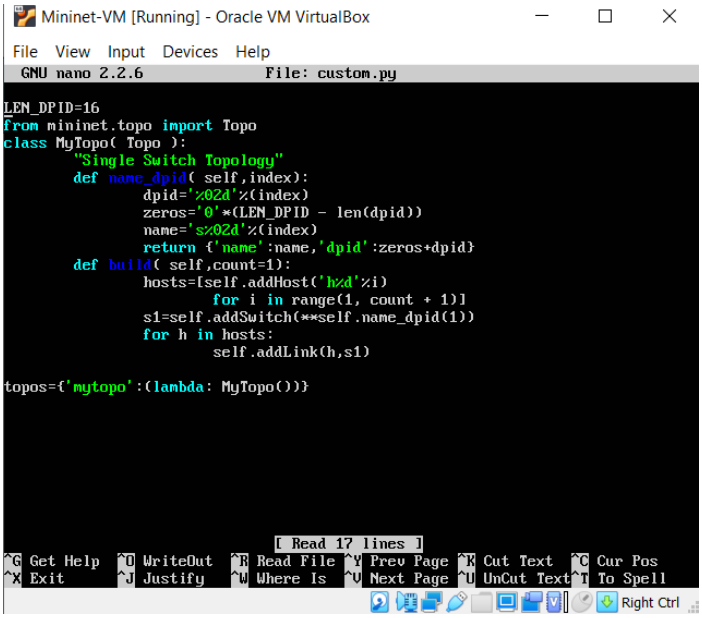

Figure 7: Custom topology

→ Now we deploy the topology as shown in figure 8.

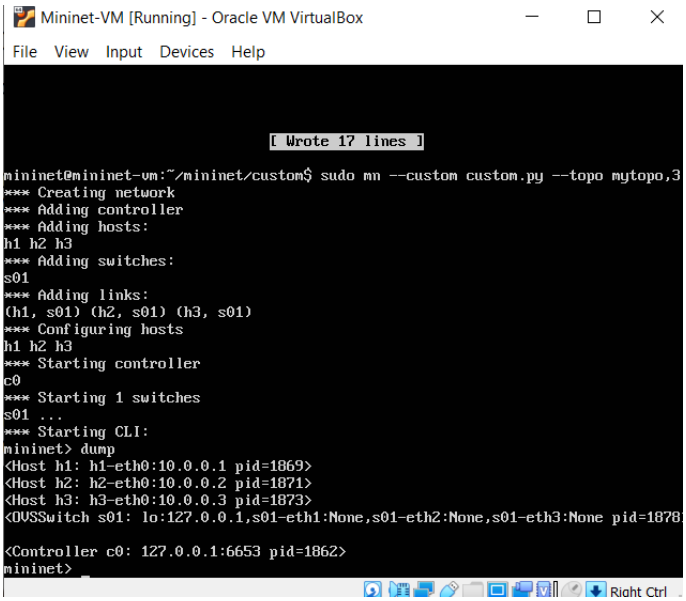

Figure 8: Deploying the custom topology