Mawlana Bhashani Science and Technology University

Lab-Report

Report No:04

Course title: Computer Network Lab

Date of Performance:21-02-21

Date of Submission:25-04-21

Submitted by

Name: Sabikun Nahar Piya

ID:IT-18020

Nusrat Jahan Jui

ID:IT-18039

3rd year 2nd semester

Session: 2017-2018

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Lab 4:SDN Controllers and Mininet

1.Objectives:

The objective of the lab 4 is to:

♣ Install and use traffic generators as powerful tools for testing network performance.

♣ Install and configure SDN Controller

♣ Install and understand how the mininet simulator works

♣ Implement and run basic examples for understanding the role of the controller and how it interact with mininet

2.Theory:

Traffic Generator:

What is iperf?: iPerf is a tool for active measurements of the maximum achievable bandwidth on IP networks. It supports tuning of various parameters related to timing, buffers and protocols (TCP, UDP, SCTP with IPv4 and IPv6). For each test it reports the bandwidth, loss, and other parameters.

Software Defined Networking: Software-defined networking was pioneered between 2008 and 2011 by work done at Stanford University and the Nicira Company (now part of VMware). The basic premise behind SDN is that by separating control of network functions from hardware devices, administrators acquire more power to route and direct traffic in response to changing requirements.

Controller: OVS-testcontroller is a simple OpenFlow controller that manages any number of switches over the OpenFlow protocol, causing them to function as L2 MAC-learning switches or hubs. It is suitable for initial testing of OpenFlow networks.

Mininet: Mininet creates a realistic virtual network, running real kernel, switch and application code, on a single machine (VM, cloud or native) Because you can easily interact with your network using the Mininet CLI (and API), customize it, share it with others, or deploy it on real hardware, Mininet is useful for

development, teaching, and research. Mininet is also a great way to develop, share, and experiment with OpenFlow and Software-Defined Networking systems.

3.Methodology:

Install iperf:

```
piya@piya-VirtualBox:~$ sudo apt-get install iperf
[sudo] password for piya:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  iperf
0 upgraded, 1 newly installed, 0 to remove and 126 not upgraded.
Need to get 76.5 kB of archives.
After this operation, 213 kB of additional disk space will be used.
Get:1 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 iperf amd64 2.0.
13+dfsg1-1build1 [76.5 kB]
Fetched 76.5 kB in 2s (41.7 kB/s)
Selecting previously unselected package iperf.
(Reading database ... 182825 files and directories currently installed.)
Preparing to unpack .../iperf_2.0.13+dfsg1-1build1_amd64.deb ...
Unpacking iperf (2.0.13+dfsg1-1build1) ...
Setting up iperf (2.0.13+dfsg1-1build1) ...
Processing triggers for man-db (2.9.1-1) ...
piya@piya-VirtualBox:~$
```

Install mininet:

```
piya@piya-VirtualBox:~$ sudo apt-get install mininet
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  cgroup-tools libcgroup1 libpython2-stdlib libpython2.7-minimal
  libpython2.7-stdlib libunbound8 net-tools openvswitch-common
  openvswitch-switch python-pkg-resources python2 python2-minimal pytho
  python2.7-minimal python3-openvswitch python3-sortedcontainers socat
Suggested packages:
  openvswitch-doc python-setuptools python2-doc python-tk python2.7-doc
  binutils binfmt-support python-sortedcontainers-doc
The following NEW packages will be installed:
  cgroup-tools libcgroup1 libpython2-stdlib libpython2.7-minimal
```

```
Reading state information... Done
The following additional packages will be installed:
  cgroup-tools libcgroup1 libpython2-stdlib libpython2.7-minimal
  libpython2.7-stdlib libunbound8 net-tools openvswitch-common
  openvswitch-switch python-pkg-resources python2 python2-minimal python2.7
  python2.7-minimal python3-openvswitch python3-sortedcontainers socat
Suggested packages:
  openvswitch-doc python-setuptools python2-doc python-tk python2.7-doc
  binutils binfmt-support python-sortedcontainers-doc
The following NEW packages will be installed:
  cgroup-tools libcgroup1 libpython2-stdlib libpython2.7-minimal
  libpython2.7-stdlib libunbound8 mininet net-tools openvswitch-common
  openvswitch-switch python-pkg-resources python2 python2-minimal python2.7
  python2.7-minimal python3-openvswitch python3-sortedcontainers socat
0 upgraded, 18 newly installed, 0 to remove and 126 not upgraded.
Need to get 7,852 kB of archives.
After this operation, 34.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://bd.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libpytho
n2.7-minimal amd64 2.7.18-1~20.04.1 [335 kB]
Get:2 http://bd.archive.ubuntu.com/ubuntu focal-updates/universe amd64 python2.
7-minimal amd64 2.7.18-1~20.04.1 [1,285 kB]
Get:3 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 python2-minimal
amd64 2.7.17-2ubuntu4 [27.5 kB]
```

```
0 upgraded, 18 newly installed, 0 to remove and 126 not upgraded.
Need to get 7,852 kB of archives.
After this operation, 34.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://bd.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libpytho
n2.7-minimal amd64 2.7.18-1~20.04.1 [335 kB]
Get:2 http://bd.archive.ubuntu.com/ubuntu focal-updates/universe amd64 python2.
7-minimal amd64 2.7.18-1~20.04.1 [1,285 kB]
Get:3 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 python2-minimal
amd64 2.7.17-2ubuntu4 [27.5 kB]
Get:4 http://bd.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libpytho
n2.7-stdlib amd64 2.7.18-1~20.04.1 [1,887 kB]
Get:5 http://bd.archive.ubuntu.com/ubuntu focal-updates/universe amd64 python2.
7 amd64 2.7.18-1~20.04.1 [248 kB]
Get:6 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 libpython2-stdli
b amd64 2.7.17-2ubuntu4 [7,072 B]
Get:7 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 python2 amd64 2.
7.17-2ubuntu4 [26.5 kB]
Get:8 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 libcgroup1 amd64
 0.41-10 [42.9 kB]
Get:9 http://bd.archive.ubuntu.com/ubuntu focal/universe amd64 cgroup-tools amd
64 0.41-10 [66.2 kB]
Get:10 http://bd.archive.ubuntu.com/ubuntu focal-updates/main amd64 libunbound8
 amd64 1.9.4-2ubuntu1.1 [349 kB]
```

```
Preparing to unpack .../2-python2-minimal_2.7.17-2ubuntu4_amd64.deb ...
Unpacking python2-minimal (2.7.17-2ubuntu4) ...
Selecting previously unselected package libpython2.7-stdlib:amd64.
Preparing to unpack .../3-libpython2.7-stdlib_2.7.18-1~20.04.1_amd64.deb ...
Unpacking libpython2.7-stdlib:amd64 (2.7.18-1~20.04.1) ...
Selecting previously unselected package python2.7.
Preparing to unpack .../4-python2.7_2.7.18-1~20.04.1_amd64.deb ...
Unpacking python2.7 (2.7.18-1~20.04.1) ...
Selecting previously unselected package libpython2-stdlib:amd64.
Preparing to unpack .../5-libpython2-stdlib_2.7.17-2ubuntu4_amd64.deb ...
Unpacking libpython2-stdlib:amd64 (2.7.17-2ubuntu4) ...
Setting up libpython2.7-minimal:amd64 (2.7.18-1~20.04.1) ...
Setting up python2.7-minimal (2.7.18-1~20.04.1) ...
Linking and byte-compiling packages for runtime python2.7...
Setting up python2-minimal (2.7.17-2ubuntu4) ...
Selecting previously unselected package python2.
(Reading database ... 183584 files and directories currently installed.)
Preparing to unpack .../00-python2_2.7.17-2ubuntu4_amd64.deb ...
Unpacking python2 (2.7.17-2ubuntu4) ...
Selecting previously unselected package libcgroup1:amd64.
Preparing to unpack .../01-libcgroup1_0.41-10_amd64.deb ...
Unpacking libcgroup1:amd64 (0.41-10) ...
Selecting previously unselected package cgroup-tools.
Preparing to unpack .../02-cgroup-tools_0.41-10_amd64.deb ...
```

```
Preparing to unpack .../11-openvswitch-switch_2.13.1-0ubuntu0.20.04.4_amd64.deb
 ...
Unpacking openvswitch-switch (2.13.1-0ubuntu0.20.04.4) ...
Setting up net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Setting up python3-sortedcontainers (2.1.0-2) ...
Setting up python3-openvswitch (2.13.1-0ubuntu0.20.04.4) ...
Setting up libpython2.7-stdlib:amd64 (2.7.18-1~20.04.1) ...
Setting up libunbound8:amd64 (1.9.4-2ubuntu1.1) ...
Setting up socat (1.7.3.3-2) ...
Setting up openvswitch-common (2.13.1-0ubuntu0.20.04.4) ...
Setting up libcgroup1:amd64 (0.41-10) ...
Setting up openvswitch-switch (2.13.1-0ubuntu0.20.04.4) ...
update-alternatives: using /usr/lib/openvswitch-switch/ovs-vswitchd to provide
/usr/sbin/ovs-vswitchd (ovs-vswitchd) in auto mode
Created symlink /etc/systemd/system/multi-user.target.wants/openvswitch-switch.
service → /lib/systemd/system/openvswitch-switch.service.
Setting up python2.7 (2.7.18-1~20.04.1) ...
Setting up libpython2-stdlib:amd64 (2.7.17-2ubuntu4) ...
Setting up cgroup-tools (0.41-10) ...
Setting up python2 (2.7.17-2ubuntu4) ...
Setting up python-pkg-resources (44.0.0-2) ...
Setting up mininet (2.2.2-5ubuntu1) ...
Processing triggers for mime-support (3.64ubuntu1) ...
Processing triggers for gnome-menus (3.36.0-1ubuntu1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
Processing triggers for systemd (245.4-4ubuntu3.4) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for desktop-file-utils (0.24-1ubuntu3) ...
piya@piya-VirtualBox:~$
```


4.Exercises:

Exercise 4.1.1: Open a Linux terminal, and execute the command line iperf --help. Provide four configuration options of iperf.

```
piya@piya-VirtualBox:~$ iperf --help
Usage: iperf [-s|-c host] [options]
       iperf [-h|--help] [-v|--version]

Client/Server:
  -b, --bandwidth #[kmgKMG | pps]  bandwidth to send at in bits/sec or packets
per second
  -e, --enhancedreports    use enhanced reporting giving more tcp/udp and traff
ic information
  -f, --format    [kmgKMG]   format to report: Kbits, Mbits, KBytes, MBytes
  -i, --interval  #        seconds between periodic bandwidth reports
  -l, --len       #[kmKM]    length of buffer in bytes to read or write (Defaul
ts: TCP=128K, v4 UDP=1470, v6 UDP=1450)
```

```
  -p, --port      #        server port to listen on/connect to
  -u, --udp                use UDP rather than TCP
      --udp-counters-64bit use 64 bit sequence numbers with UDP
  -w, --window    #[KM]    TCP window size (socket buffer size)
  -z, --realtime           request realtime scheduler
  -B, --bind <host>[:<port>][%<dev>] bind to <host>, ip addr (including multica
st address) and optional port and device
  -C, --compatibility      for use with older versions does not sent extra msgs
  -M, --mss       #        set TCP maximum segment size (MTU - 40 bytes)
  -N, --nodelay            set TCP no delay, disabling Nagle's Algorithm
  -S, --tos       #        set the socket's IP_TOS (byte) field

Server specific:
  -s, --server             run in server mode
  -t, --time      #        time in seconds to listen for new connections as wel
l as to receive traffic (default not set)
      --udp-histogram #,#  enable UDP latency histogram(s) with bin width and c
ount, e.g. 1,1000=1(ms),1000(bins)
  -B, --bind <ip>[%<dev>]  bind to multicast address and optional device
  -H, --ssm-host <ip>      set the SSM source, use with -B for (S,G)
  -U, --single_udp         run in single threaded UDP mode
  -D, --daemon             run the server as a daemon
  -V, --ipv6_domain        Enable IPv6 reception by setting the domain and sock
et to AF_INET6 (Can receive on both IPv4 and IPv6)
```

Exercise 4.1.2: Open two Linux terminals, and configure terminal-1 as client (iperf –c IPv4_server_address) and terminal-2 as server (iperf -s).

For terminal-1:

```
piya@piya-VirtualBox:~$ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
```

For terminal-2:

```
piya@piya-VirtualBox:~$ iperf -c 127.0.0.1 -u
------------------------------------------------------------
Client connecting to 127.0.0.1, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 127.0.0.1 port 45019 connected with 127.0.0.1 port 5001
read failed: Connection refused
[  3] WARNING: did not receive ack of last datagram after 1 tries.
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 891 datagrams
piya@piya-VirtualBox:~$
```

Exercise 4.1.3: Open two Linux terminals, and configure terminal-1 as client and terminal-2 as server for exchanging UDP traffic, which are the command lines? Which are the statistics are provided at the end of transmission? What is different from the statistics provided in exercise 4.1.1

```
piya@piya-VirtualBox:~$ iperf -c 127.0.0.1 -u
------------------------------------------------------------
Client connecting to 127.0.0.1, UDP port 5001
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 127.0.0.1 port 45767 connected with 127.0.0.1 port 5001
read failed: Connection refused
[  3] WARNING: did not receive ack of last datagram after 1 tries.
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[  3] Sent 891 datagrams
```

```
piya@piya-VirtualBox:~$ iperf -s -u
------------------------------------------------------------
Server listening on UDP port 5001
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
```

Exercise 4.1.4: Open two Linux terminals, and configure terminal-1 as client and terminal-2 as server for exchanging UDP traffic, with:

o Packet length = 1000bytes

o Time = 20 seconds

o Bandwidth = 1Mbps

o Port = 9900

Which are the command lines?

For terminal 1:

```
piya@piya-VirtualBox:~$ iperf -c 127.0.0.1 -u -l 1000 -t 20 -b 1 -p 9900
WARNING: delay too large, reducing from 8000.0 to 1.0 seconds.
------------------------------------------------------------
Client connecting to 127.0.0.1, UDP port 9900
Sending 1000 byte datagrams, IPG target: 8000000000.00 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 127.0.0.1 port 37270 connected with 127.0.0.1 port 9900
read failed: Connection refused
[  3] WARNING: did not receive ack of last datagram after 2 tries.
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-20.0 sec  19.5 KBytes  8.00 Kbits/sec
[  3] Sent 20 datagrams
piya@piya-VirtualBox:~$
```

For terminal 2:

```
piya@piya-VirtualBox:~$ iperf -s -u -p 9900
------------------------------------------------------------
Server listening on UDP port 9900
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
```

4.2:Using mininet

Exercise 4.2.1: Open two Linux terminals, and execute the command line ifconfig in terminal-1. How many interfaces are present?

In terminal-2, execute the command line sudo mn, which is the output?

In terminal-1 execute the command line ifconfig. How many real and virtual interfaces are present now?

```
piya@piya-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::b977:b09c:b4c1:bdcd  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:95:5c:6c  txqueuelen 1000  (Ethernet)
        RX packets 7361  bytes 10280095 (10.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3183  bytes 242277 (242.2 KB)
        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 4022  bytes 3771003 (3.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4022  bytes 3771003 (3.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

piya@piya-VirtualBox:~$
```

```
piya@piya-VirtualBox:~$ sudo mn
[sudo] password for piya:
*** No default OpenFlow controller found for default switch!
*** Falling back to OVS Bridge
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller

*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

```
piya@piya-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::b977:b09c:b4c1:bdcd  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:95:5c:6c  txqueuelen 1000  (Ethernet)
        RX packets 7379  bytes 10282192 (10.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3201  bytes 243698 (243.6 KB)
        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 4050  bytes 3773327 (3.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4050  bytes 3773327 (3.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

piya@piya-VirtualBox:~$
```

Exercise 4.2.2: Interacting with mininet; in terminal-2, display the following command lines and explain what it does:

mininet> help

```
mininet> help

Documented commands (type help <topic>):
========================================
EOF     gterm   iperfudp  nodes         pingpair      py       switch
dpctl   help    link       noecho        pingpairfull  quit     time
dump    intfs   links      pingall       ports         sh       x
exit    iperf   net        pingallfull   px            source   xterm

You may also send a command to a node using:
  <node> command {args}
For example:
  mininet> h1 ifconfig

The interpreter automatically substitutes IP addresses
for node names when a node is the first arg, so commands
like
  mininet> h2 ping h3
should work.

Some character-oriented interactive commands require
noecho:
  mininet> noecho h2 vi foo.py
However, starting up an xterm/gterm is generally better:
  mininet> xterm h2
```

mininet> nodes

```
mininet> nodes
available nodes are:
h1 h2 s1
```

mininet> net

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
```

mininet> dump

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=5840>
<Host h2: h2-eth0:10.0.0.2 pid=5842>
<OVSBridge s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=5847>
```

mininet> h1 ifconfig –a

```
mininet> h1 ifconfig -a
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::50e3:98ff:fe84:a384  prefixlen 64  scopeid 0x20<link>
        ether 52:e3:98:84:a3:84  txqueuelen 1000  (Ethernet)
        RX packets 40  bytes 4397 (4.3 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  collisio

mininet>
```


mininet> s1 ifconfig –a

```
mininet> s1 ifconfig -a
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::b977:b09c:b4c1:bdcd  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:95:5c:6c  txqueuelen 1000  (Ethernet)
        RX packets 7411  bytes 10285787 (10.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3235  bytes 246459 (246.4 KB)
        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 4078  bytes 3775763 (3.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4078  bytes 3775763 (3.7 MB)
```



```
        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

s1: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 02:1c:b6:44:05:4c  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 22  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::f42d:afff:febb:c7a5  prefixlen 64  scopeid 0x20<link>
        ether f6:2d:af:bb:c7:a5  txqueuelen 1000  (Ethernet)
        RX packets 13  bytes 1006 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 43  bytes 4662 (4.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::b030:b4ff:fe25:ec47  prefixlen 64  scopeid 0x20<link>
        ether b2:30:b4:25:ec:47  txqueuelen 1000  (Ethernet)
        RX packets 13  bytes 1006 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 41  bytes 4486 (4.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisio

mininet>
```


mininet> h1 ping –c 5 h2

```
mininet> h1 ping -c 5 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=0.481 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.128 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.125 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.128 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.125 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4103ms
rtt min/avg/max/mdev = 0.125/0.197/0.481/0.141 ms
mininet>
```

Exercise 4.2.3:

In terminal-2, display the following command line: sudo mn --link tc,bw=10,delay=500ms

mininet> h1 ping -c 5 h2, What happen with the link?

mininet> h1 iperf –s –u &

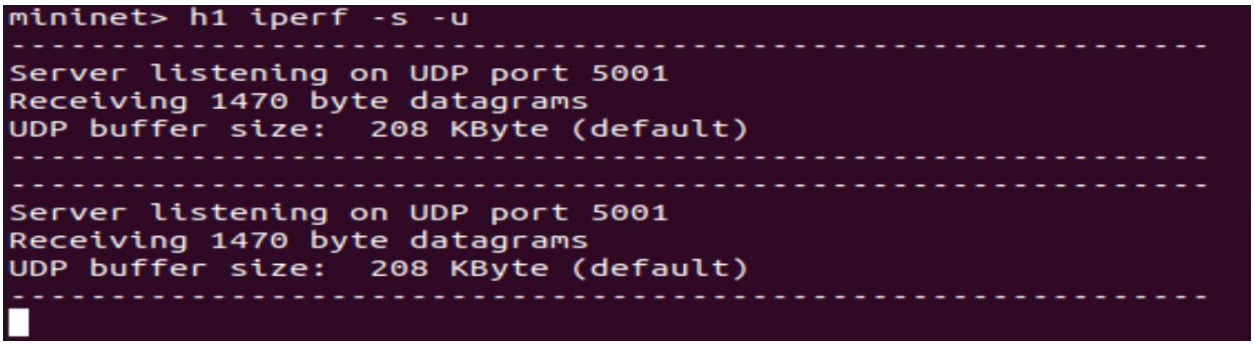
mininet> h2 iperf –c IPv4_h1 -u, Is there any packet loss? Modify iperf for creating packet loss in the mininet network, which is the command line?

```
piya@piya-VirtualBox:~$ sudo mn --link tc,bw=10,delay=500ms
[sudo] password for piya:
*** No default OpenFlow controller found for default switch!
*** Falling back to OVS Bridge
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(10.00Mbit 500ms delay) (10.00Mbit 500ms delay) (h1, s1) (10.00Mbit 500ms delay
) (10.00Mbit 500ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller

*** Starting 1 switches
s1 ...(10.00Mbit 500ms delay) (10.00Mbit 500ms delay)
*** Starting CLI:
mininet>
```

```
mininet> h1 ping -c 5 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=4003 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=3002 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=2001 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=2001 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=2000 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4049ms
rtt min/avg/max/mdev = 2000.485/2601.553/4002.835/800.772 ms,
mininet>
```



5.Conclusion:

Mininet is a network emulator which creates a network of virtual hosts, switches, controllers, and links. Mininet hosts run standard Linux network software, and its switches support OpenFlow for highly flexible custom routing and Software-Defined Networking.

Mininet's code is almost entirely Python, except for a short C utility.Mininet-based networks cannot (currently) exceed the CPU or bandwidth available on a single server.Mininet cannot (currently) run non-Linux-compatible OpenFlow switches or applications; this has not been a major issue in practice.