Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 04

Course Code:ICT-3207

Course title: Computer Network Lab

Date of Performance:30-01-2021

Date of Submission:05-02-2021

Submitted by

Name: Meraz Ahmed

ID:IT-18005

3rd year 2nd semester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Objectives : The main objectives of the lab how 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.

Theory :

Traffic Generator:

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. Software Defined

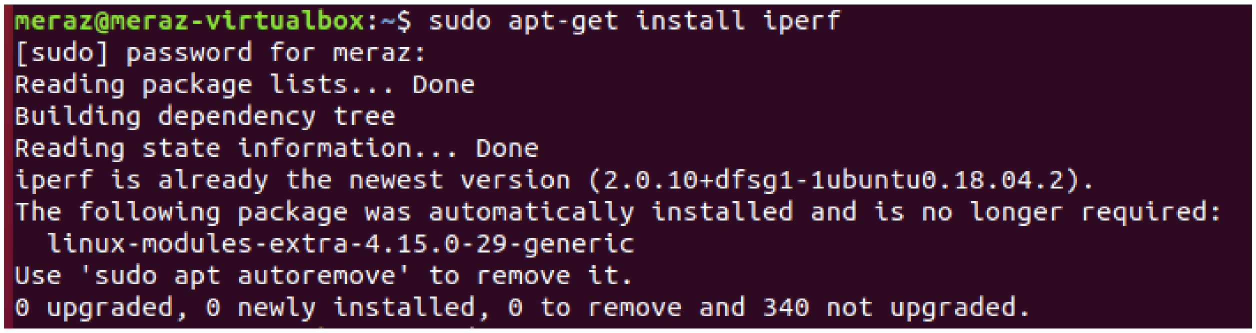
Networking: Software Defined Networking that by separating control of network functions from hardware devices, administrators acquire more power to route and direct traffic in response to changing requirements.

Controller: Controller is suitable for initial testing of OpenFlow networks. OVStestcontroller is a simple OpenFlow controller that manages any number of switches over the OpenFlow protocol, causing them to function as L2 MAClearning switches or hubs.

Mininet : Mininet creates a realistic virtual network, running real kernel, switch and application code, on a single machine .

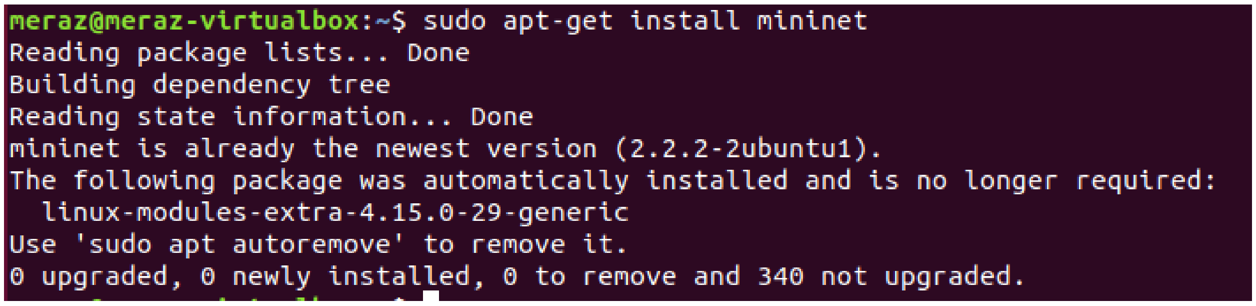
Methodology :

Install iperf:



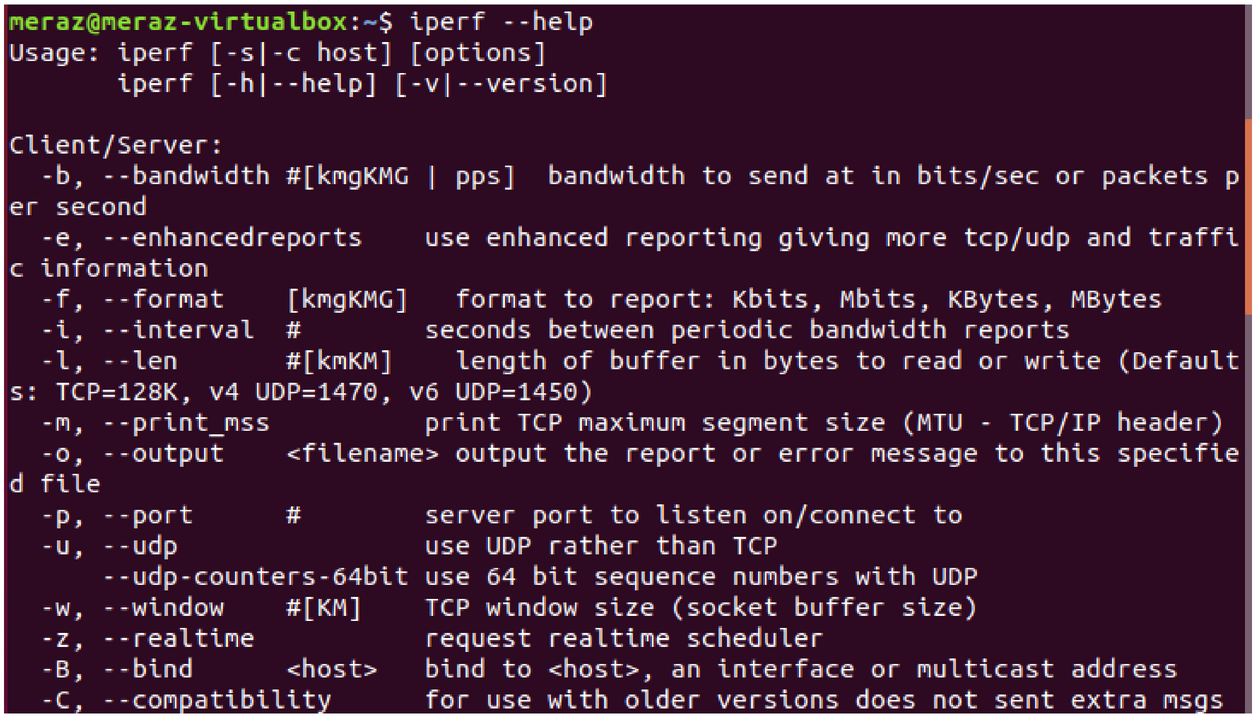
```
meraz@meraz-virtualbox:~$ sudo apt-get install iperf
[sudo] password for meraz:
Reading package lists... Done
Building dependency tree
Reading state information... Done
iperf is already the newest version (2.0.10+dfsg1-1ubuntu0.18.04.2).
The following package was automatically installed and is no longer required:
  linux-modules-extra-4.15.0-29-generic
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 340 not upgraded.
```

Install Mininet:



```
meraz@meraz-virtualbox:~$ sudo apt-get install mininet
Reading package lists... Done
Building dependency tree
Reading state information... Done
mininet is already the newest version (2.2.2-2ubuntu1).
The following package was automatically installed and is no longer required:
  linux-modules-extra-4.15.0-29-generic
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 340 not upgraded.
```

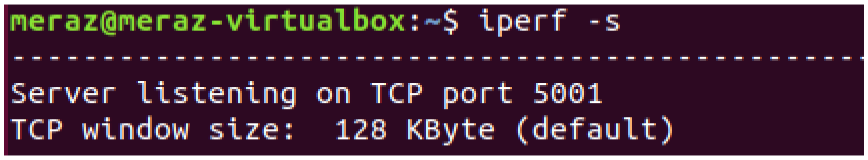
Exercises: 4.1.1: Open a Linux terminal, and execute the command line iperf --help. Provide four configuration options of iperf.



```
meraz@meraz-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 p
er second
  -e, --enhancedreports    use enhanced reporting giving more tcp/udp and traffi
c 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 (Default
s: TCP=128K, v4 UDP=1470, v6 UDP=1450)
  -m, --print_mss          print TCP maximum segment size (MTU - TCP/IP header)
  -o, --output    <filename> output the report or error message to this specifie
d file
  -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>   bind to <host>, an interface or multicast address
  -C, --compatibility      for use with older versions does not sent extra msgs
```

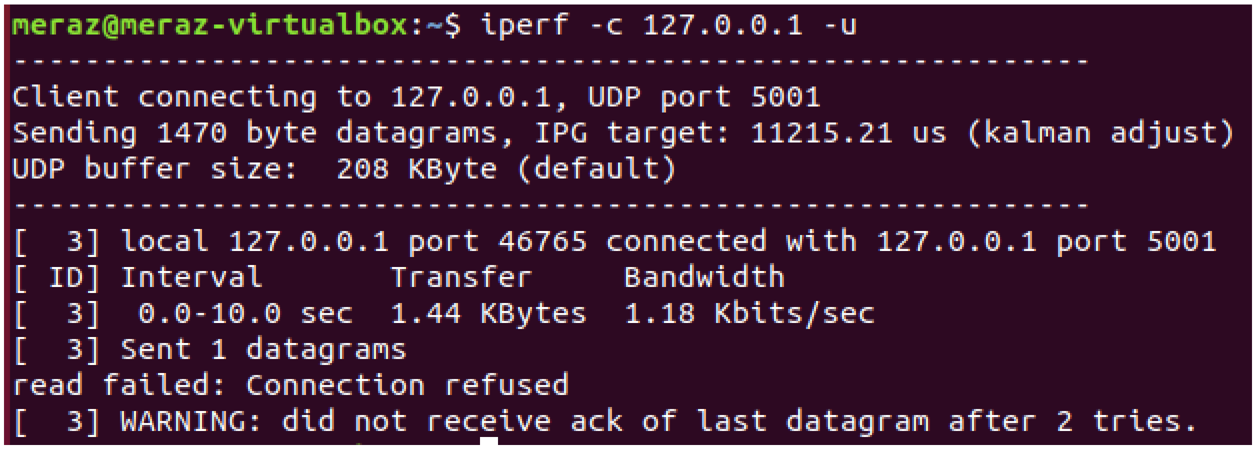
Exercise 4.1.2: Open two Linux terminals, and configure terminal-1 as client (iperf –c IPv4_server_address) and terminal-2 as server (iperf -s).

Terminal -1 :



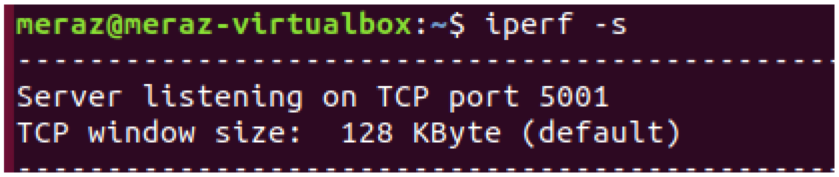
```
meraz@meraz-virtualbox:~$ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
```

Terminal -2:



```
meraz@meraz-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 46765 connected with 127.0.0.1 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec  1.44 KBytes  1.18 Kbits/sec
[  3] Sent 1 datagrams
read failed: Connection refused
[  3] WARNING: did not receive ack of last datagram after 2 tries.
```

Terminal-2 as server :



```
meraz@meraz-virtualbox:~$ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 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:

# Packet length = 1000bytes

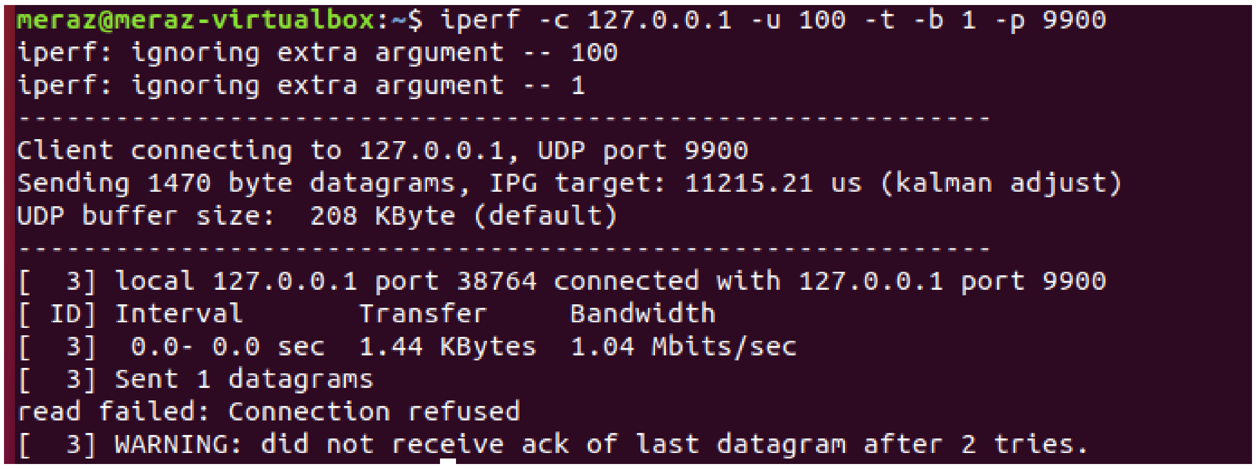
 # Time = 20 seconds

 # Bandwidth = 1Mbps

Port = 9900

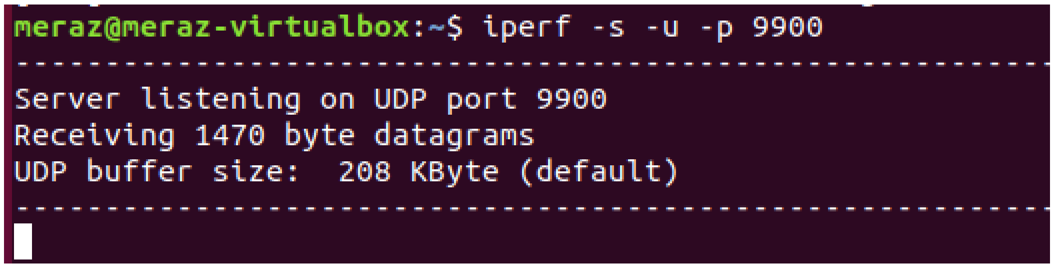
The Command lines are :

Terminal-1:



```
meraz@meraz-virtualbox:~$ iperf -c 127.0.0.1 -u 100 -t -b 1 -p 9900
iperf: ignoring extra argument -- 100
iperf: ignoring extra argument -- 1
------------------------------------------------------------
Client connecting to 127.0.0.1, UDP port 9900
Sending 1470 byte datagrams, IPG target: 11215.21 us (kalman adjust)
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[  3] local 127.0.0.1 port 38764 connected with 127.0.0.1 port 9900
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 0.0 sec  1.44 KBytes  1.04 Mbits/sec
[  3] Sent 1 datagrams
read failed: Connection refused
[  3] WARNING: did not receive ack of last datagram after 2 tries.
```

Terminal-2 :

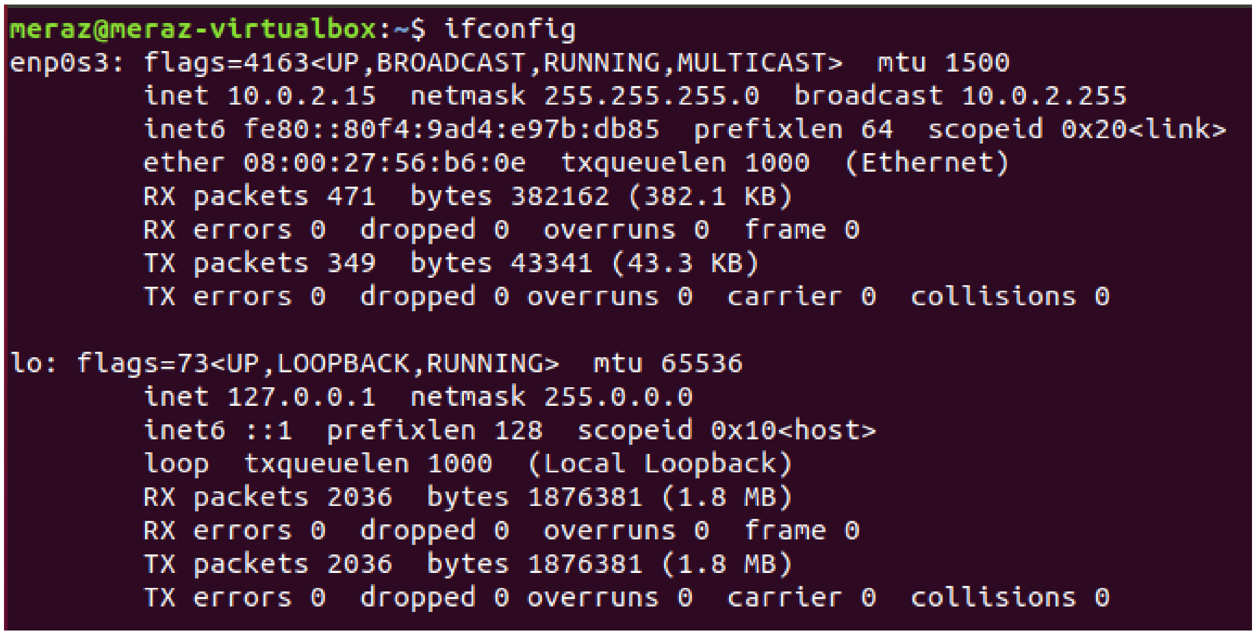


```
meraz@meraz-virtualbox:~$ iperf -s -u -p 9900
------------------------------------------------------------
Server listening on UDP port 9900
Receiving 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
```

Using Mininet:

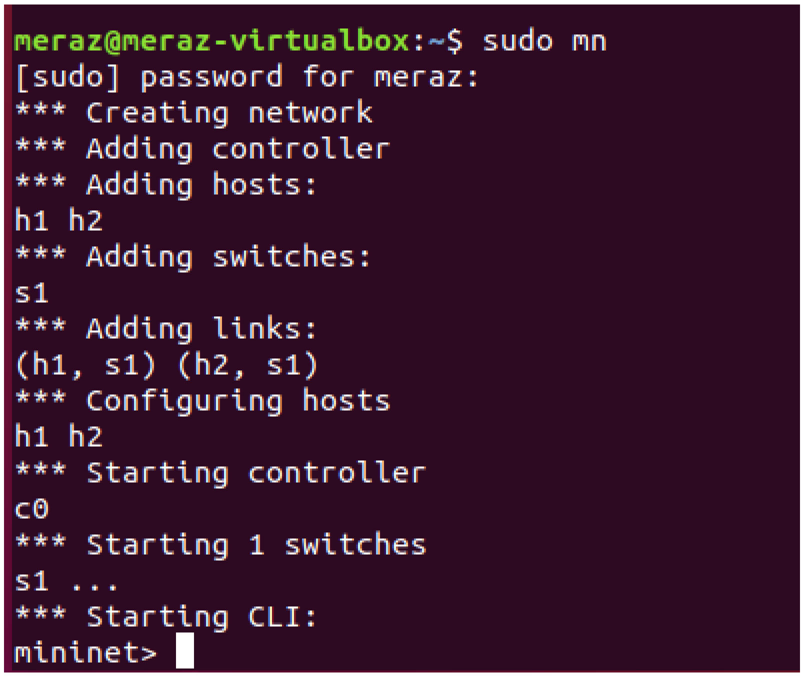
Exercise 4.2.1: Open two Linux terminals, and execute the command line ifconfig in terminal-1.

Interfaces are :



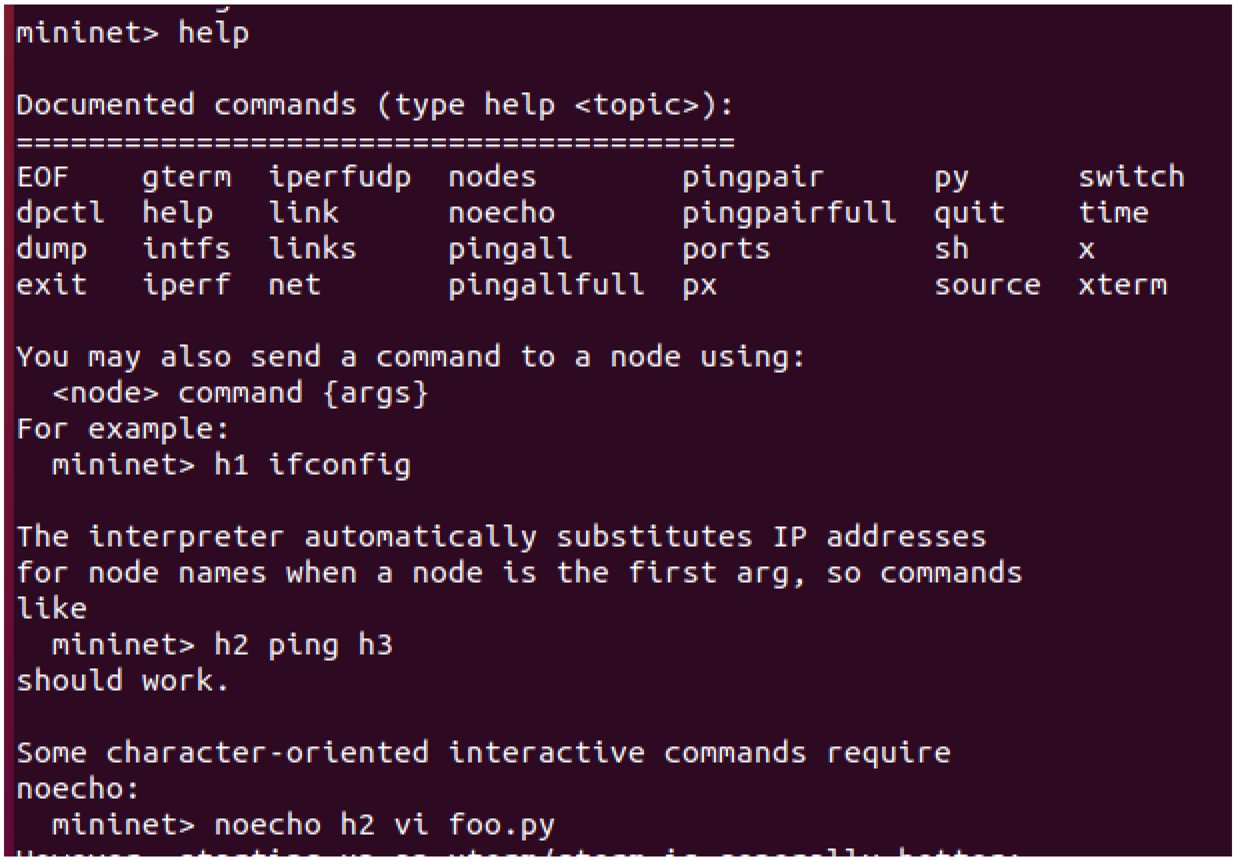
```
meraz@meraz-virtualbox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::80f4:9ad4:e97b:db85  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:56:b6:0e  txqueuelen 1000  (Ethernet)
        RX packets 471  bytes 382162 (382.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 349  bytes 43341 (43.3 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 2036  bytes 1876381 (1.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2036  bytes 1876381 (1.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

In terminal-2, execute the command line sudo mn:


```
meraz@meraz-virtualbox:~$ sudo mn
[sudo] password for meraz:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

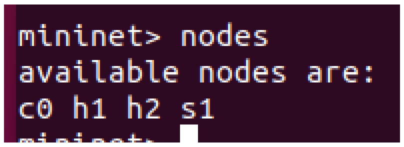
Exercise 4.2.2: Interacting with mininet; in terminal-2, display the following command lines and explain what it does:

# mininet> help:



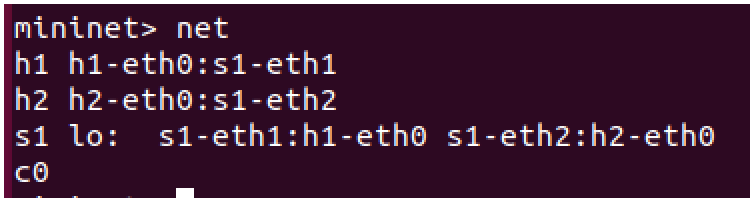
```
mininet> help

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
```

# mininet> nodes:



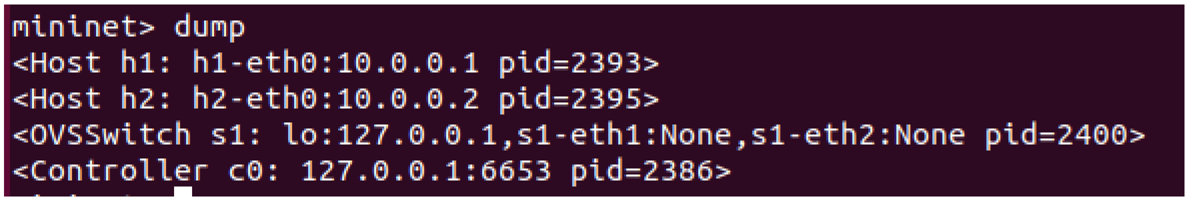
```
mininet> nodes
available nodes are:
c0 h1 h2 s1
```

# mininet> net:



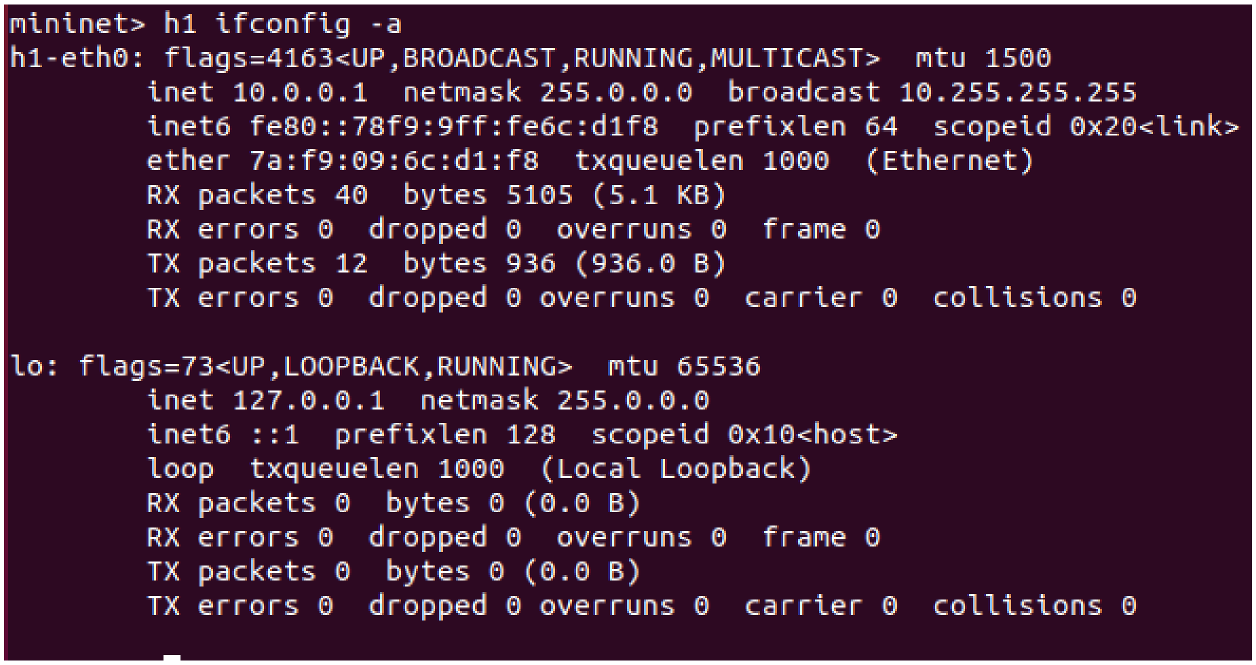
```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
```

# mininet> dump:



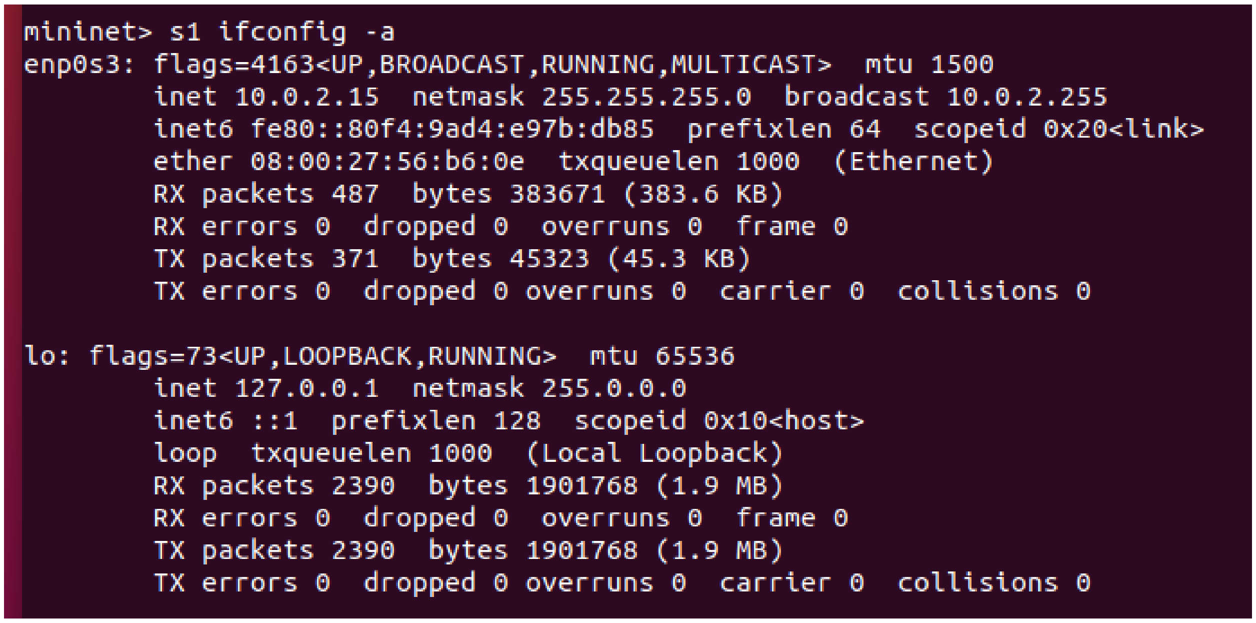
```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2393>
<Host h2: h2-eth0:10.0.0.2 pid=2395>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2400>
<Controller c0: 127.0.0.1:6653 pid=2386>
```

# 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::78f9:9ff:fe6c:d1f8  prefixlen 64  scopeid 0x20<link>
        ether 7a:f9:09:6c:d1:f8  txqueuelen 1000  (Ethernet)
        RX packets 40  bytes 5105 (5.1 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> 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::80f4:9ad4:e97b:db85  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:56:b6:0e  txqueuelen 1000  (Ethernet)
        RX packets 487  bytes 383671 (383.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 371  bytes 45323 (45.3 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 2390  bytes 1901768 (1.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2390  bytes 1901768 (1.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```