



# **Mawlana Bhashani Science and Technology University**

## **Lab-Report**

Course Title: Computer Network Lab

Lab Report No: 04

Lab Report Name: SDN Controllers and Mininet

### **Submitted by**

Name: Ali Ashadullah Arif &  
Ahadul Haque  
ID: IT-18031 & IT-18045  
3<sup>rd</sup> Year 2<sup>nd</sup> Semester  
Session: 2017-2018  
Dept. of ICT  
MBSTU.

### **Submitted To**

Nazrul Islam  
Assistant Professor  
Dept. of ICT  
MBSTU.

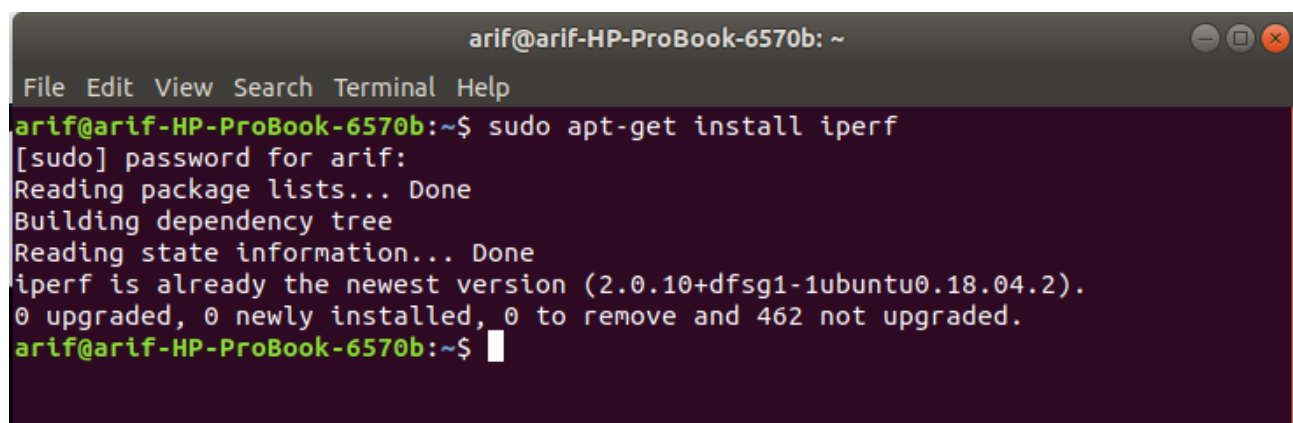
## Theory:

### Traffic Generator:

**iPerf** : 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. Iperf allows the user to set various parameters that can be used for testing a network, or alternatively for optimizing or tuning a network. 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.

**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.

### Install iPerf

A terminal window titled 'arif@arif-HP-ProBook-6570b: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command 'sudo apt-get install iperf' being executed. The output indicates that iperf is already installed at the latest version (2.0.10+dfsg1-1ubuntu0.18.04.2) and no action is needed. The prompt returns to 'arif@arif-HP-ProBook-6570b:~\$'.

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ sudo apt-get install iperf  
[sudo] password for arif:  
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).  
0 upgraded, 0 newly installed, 0 to remove and 462 not upgraded.  
arif@arif-HP-ProBook-6570b:~$
```

## Install Mininet

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ sudo apt-get install mininet  
[sudo] password for arif:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
mininet is already the newest version (2.2.2-2ubuntu1).  
0 upgraded, 0 newly installed, 0 to remove and 462 not upgraded.  
arif@arif-HP-ProBook-6570b:~$
```

## 4. Exercise Using iperf

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

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ 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 traffic 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 (Defaults: TCP=128K, v4 UDP=128K)  
-m, --print_mss print TCP maximum segment size (MTU - TCP/IP header)  
-o, --output <filename> output the report or error message to this specified 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  
-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 well as to receive traffic  
-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 socket to AF_INET6 (Can receive  
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)

**Terminal-1:**

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ iperf -s  
-----  
Server listening on TCP port 5001  
TCP window size: 128 KByte (default)  
-----  
█
```

**Terminal-2:**

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ 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 56923 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.  
arif@arif-HP-ProBook-6570b:~$
```

**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?

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ 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 56923 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.  
arif@arif-HP-ProBook-6570b:~$
```

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ 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:

Packet length = 1000bytes, Time = 20 seconds, Bandwidth = 1Mbps,  
Port = 9900

**Which are the command lines?**

**The command lines are:**

**For terminal 1:**

`iperf -c 127.0.0.1 -u -l 1000 -t 20 -b 1 -p 9900`

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ 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 53953 connected with 127.0.0.1 port 9900  
[ ID] Interval      Transfer    Bandwidth  
[ 3] 0.0-20.0 sec 1000 Bytes  400 bits/sec  
[ 3] Sent 1 datagrams  
read failed: Connection refused  
[ 3] WARNING: did not receive ack of last datagram after 2 tries.  
arif@arif-HP-ProBook-6570b:~$
```

**For terminal 2:**

`iperf -s -u -p 9900`

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ iperf -s -u  
-----  
Server listening on UDP port 5001  
Receiving 1470 byte datagrams  
UDP buffer size: 208 KByte (default)  
-----
```

## Exercise Using Mininet

**Exercise 4.2.1:** Open two Linux terminals, and execute the command line `ifconfig` in terminal1. 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?

```
arif@arif-HP-ProBook-6570b: ~  
File Edit View Search Terminal Help  
arif@arif-HP-ProBook-6570b:~$ ifconfig  
enp0s25: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
        ether d8:9d:67:c8:dc:9e txqueuelen 1000 (Ethernet)  
        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  
        device interrupt 17 memory 0xd4700000-d4720000  
  
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 4108 bytes 3783394 (3.7 MB)  
        RX errors 0 dropped 0 overruns 0 frame 0  
        TX packets 4108 bytes 3783394 (3.7 MB)  
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
        inet 192.168.43.142 netmask 255.255.255.0 broadcast 192.168.43.255  
        inet6 fe80::e494:8eae:22b7:580 prefixlen 64 scopeid 0x20<link>  
        ether 20:16:d8:bd:75:4d txqueuelen 1000 (Ethernet)  
        RX packets 345493 bytes 492561007 (492.5 MB)  
        RX errors 0 dropped 0 overruns 0 frame 0  
        TX packets 179964 bytes 17856393 (17.8 MB)  
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
arif@arif-HP-ProBook-6570b:~$
```



```
arif@arif-HP-ProBook-6570b: ~
File Edit View Search Terminal Help
arif@arif-HP-ProBook-6570b:~$ sudo mn
[sudo] password for arif:
*** 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>
```

**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     intfz  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>
```

mininet> nodes

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

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> 
```

mininet> dump

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2573>
<Host h2: h2-eth0:10.0.0.2 pid=2576>
<OVSBridge s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2581>
mininet> 
```

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::e488:dff:fe52:dcf6 prefixlen 64 scopeid 0x20<link>
    ether e6:88:0d:52:dc:f6 txqueuelen 1000 (Ethernet)
    RX packets 41 bytes 4361 (4.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 13 bytes 1006 (1.0 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 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> 
```

mininet> s1 ifconfig -a



File Edit View Search Terminal Help

```
mininet> s1 ifconfig -a
enp0s25: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether d8:9d:67:c8:dc:9e txqueuelen 1000 (Ethernet)
    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
    device interrupt 17 memory 0xd4700000-d4720000

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 203 bytes 17087 (17.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 203 bytes 17087 (17.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

ovs-system: flags=4098<BROADCAST,MULTICAST> mtu 1500
    ether da:b9:de:4a:8d:03 txqueuelen 1000 (Ethernet)
    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 fa:8e:e5:20:01:48 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 26 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::ccff:a7ff:fef9:3ada prefixlen 64 scopeid 0x20<link>
    ether ce:ff:a7:f9:3a:da txqueuelen 1000 (Ethernet)
    RX packets 13 bytes 1006 (1.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
```

```
s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet6 fe80::14b7:3eff:fed8:8e1a prefixlen 64 scopeid 0x20<link>
    ether 16:b7:3e:d8:8e:1a txqueuelen 1000 (Ethernet)
    RX packets 13 bytes 1006 (1.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 42 bytes 4448 (4.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.43.142 netmask 255.255.255.0 broadcast 192.168.43.255
    inet6 fe80::e494:8eae:22b7:580 prefixlen 64 scopeid 0x20<link>
    ether 20:16:d8:bd:75:4d txqueuelen 1000 (Ethernet)
    RX packets 1300 bytes 1570244 (1.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1178 bytes 133806 (133.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

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=0.768 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.126 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.120 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.079 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.095 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4080ms
rtt min/avg/max/mdev = 0.079/0.237/0.768/0.266 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?

```
arif@arif-HP-ProBook-6570b: ~
File Edit View Search Terminal Help
arif@arif-HP-ProBook-6570b:~$ sudo mn --link tc,bw=10,delay=500ms
[sudo] password for arif:
*** 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=2 ttl=64 time=2988 ms
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=4001 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=2000 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=2000 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 4060ms
rtt min/avg/max/mdev = 2000.154/2598.192/4001.674/799.350 ms, pipe 4
mininet> █
```

**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 supports research, development, learning, prototyping, testing, debugging, and any other tasks that could benefit from having a complete experimental network on a laptop or other PC. Mininet provides an easy way to get correct system behaviour (and, to the extent supported by your hardware, performance) and to experiment with topologies.

Mininet networks run real code including standard Unix/Linux network applications as well as the real Linux kernel and network stack (including any kernel extensions which you may have available, as long as they are compatible with network namespaces.)

Because of this, the code you develop and test on Mininet, for an OpenFlow controller, modified switch, or host, can move to a real system with minimal changes, for real-world testing, performance evaluation, and deployment. Importantly this means that a design that works in Mininet can usually move directly to hardware switches for line-rate packet forwarding.