Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 01

Course code: ICT- 3208

Course title: Computer Network Lab

Date of Performance: 10/10/2020

Date of Submission: 17/10/2020

Submitted by

Name : Amran Hossen Mamun
ID : IT-18018
3rd year 2st semester
Session : 2017 – 2018
Dept. of ICT
MBSTU

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

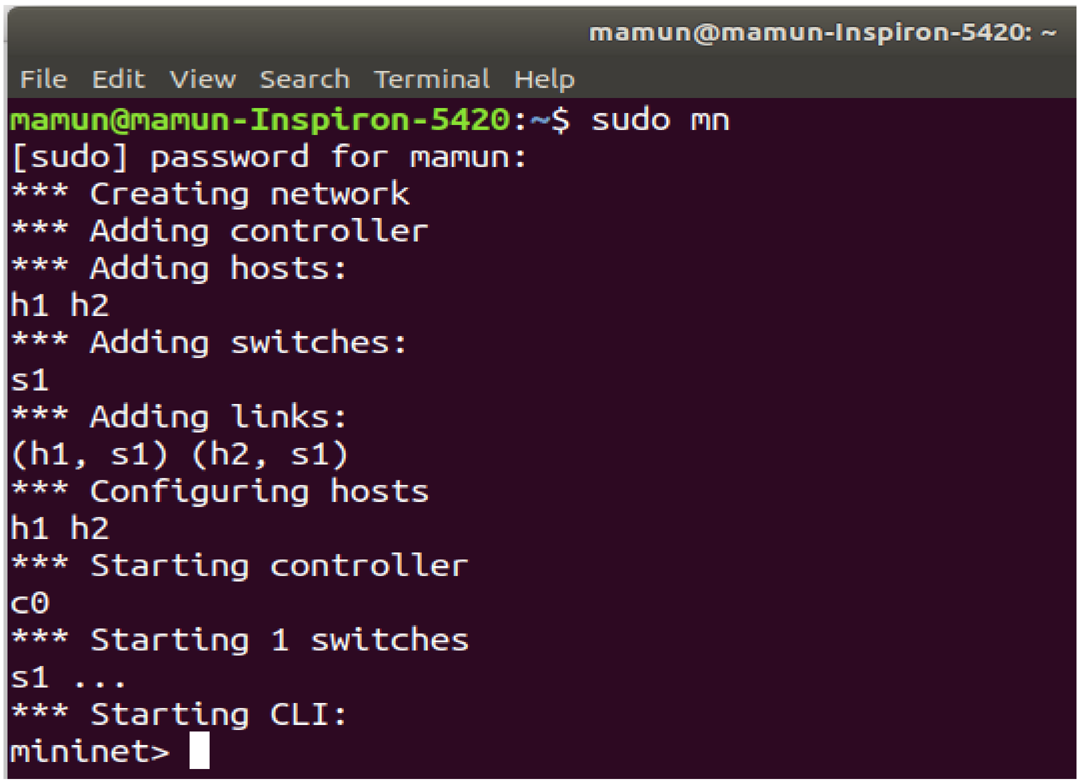
MBSTU.

Lab report no : 01

Lab report name: Basic mininet commands.

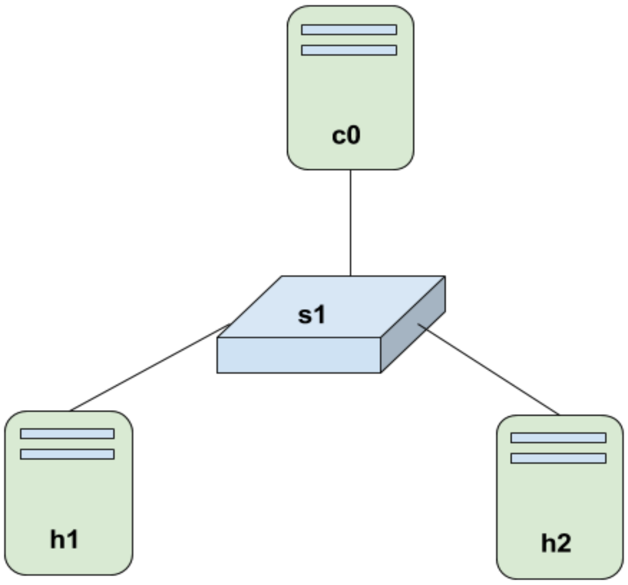
Create Virtual Network:

We will be using CLI(**sudo mn command**) to manage our virtual network. The default topology includes two hosts (h1,h2), OpenFlow Switch(s1) and OpenFlow controller(c0).

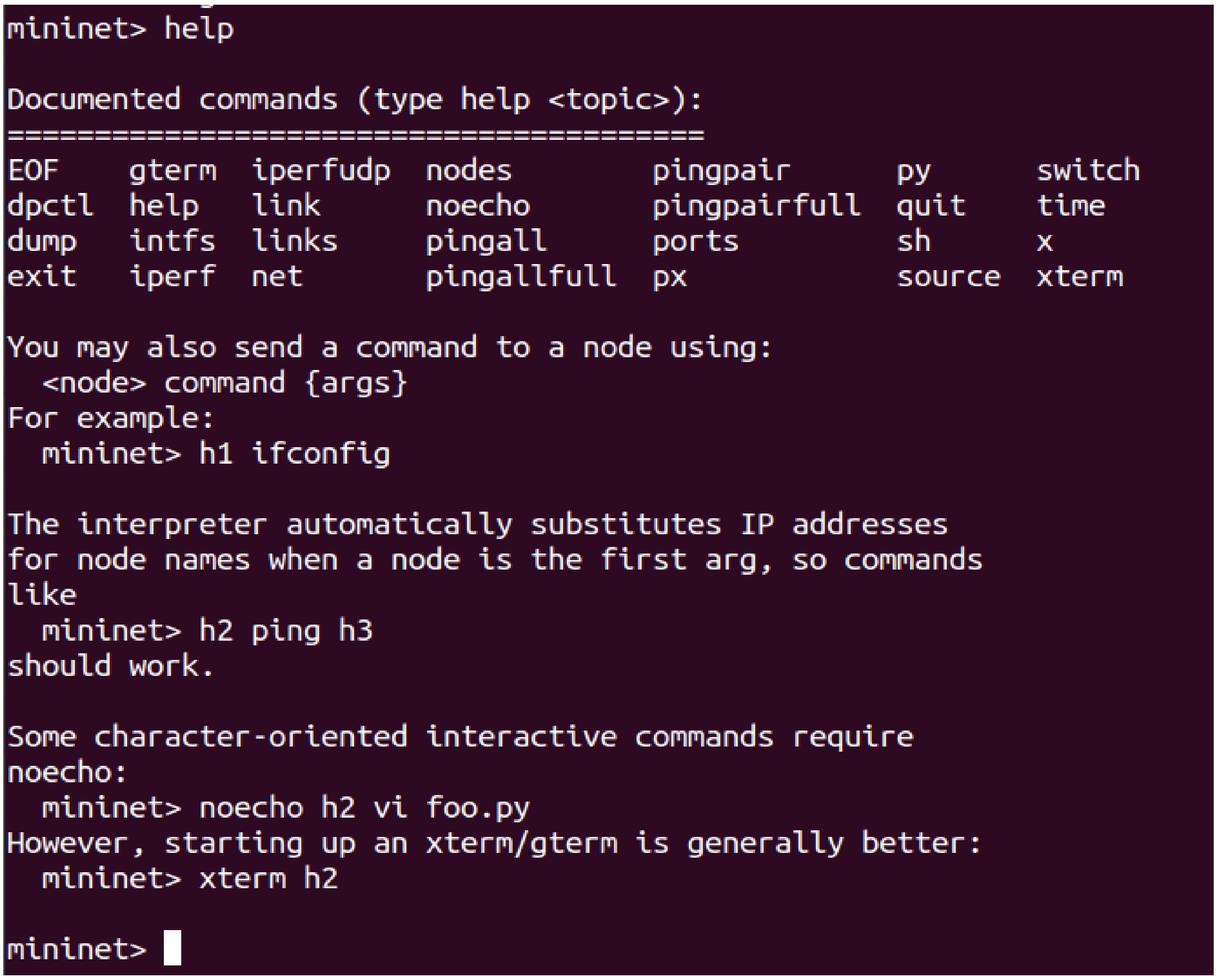

mamun@mamun-Inspiron-5420: ~
File Edit View Search Terminal Help

```
mamun@mamun-Inspiron-5420:~$ sudo mn
[sudo] password for mamun:
*** 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>
```

This command launches a simple 2 host, 1 controller, 1 switch topology. The following diagram outlines the topology.

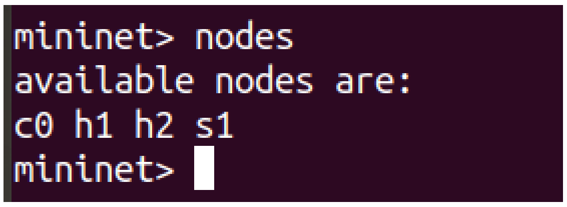


<u>Help command</u>: Run the help option to view the list of commands available:



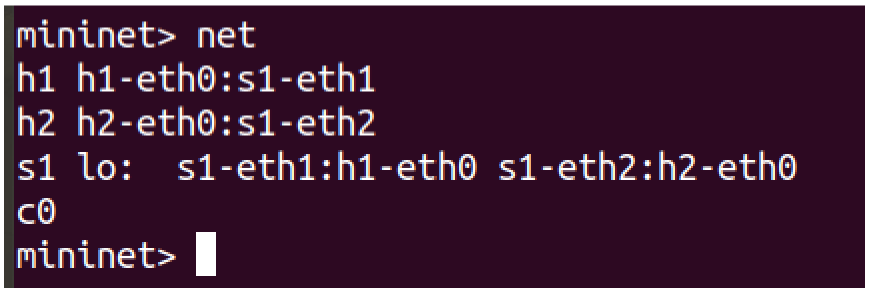
```
mininet> help

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 command: Shows the nodes we created with the simple Mininet command:



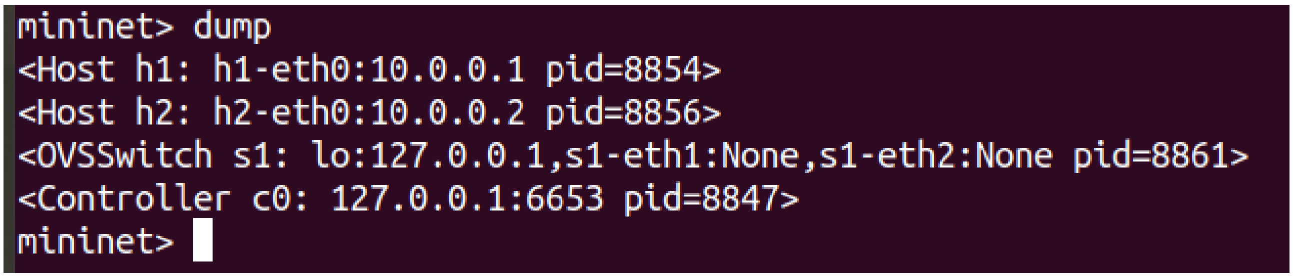
```
mininet> nodes
available nodes are:
c0 h1 h2 s1
mininet> 
```

net command: This command shows the links of all nodes



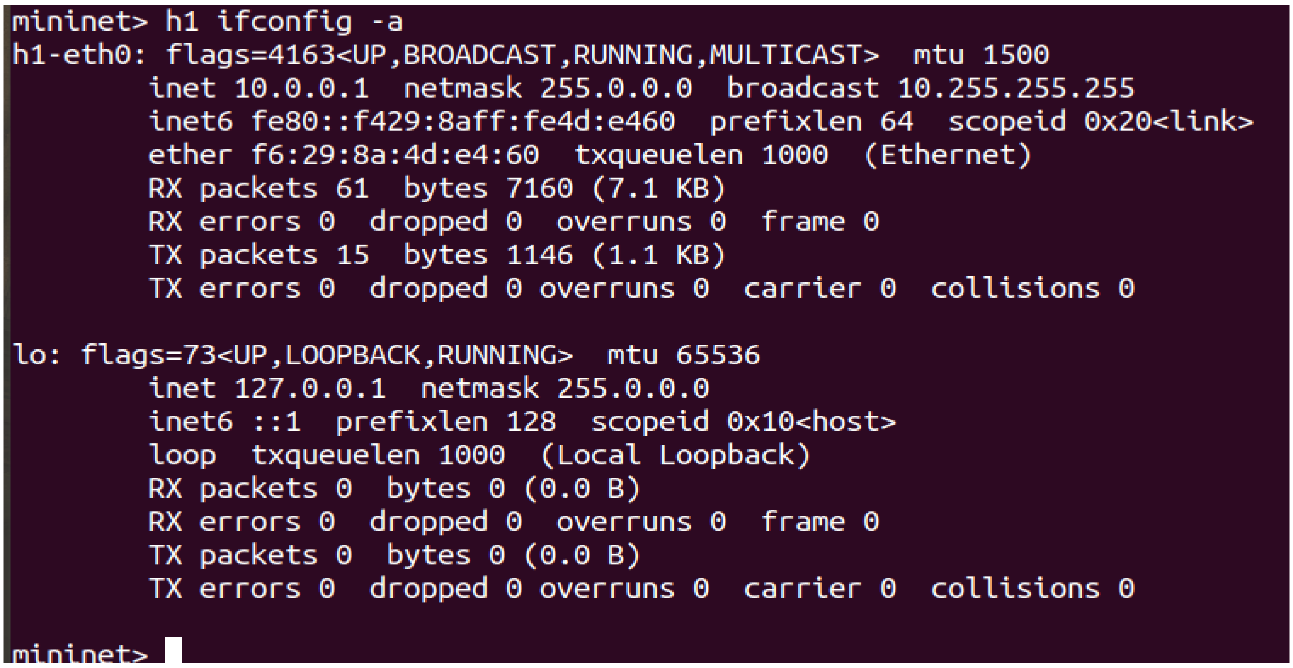
```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet> 
```

dump command: which displays summary information about all devices:



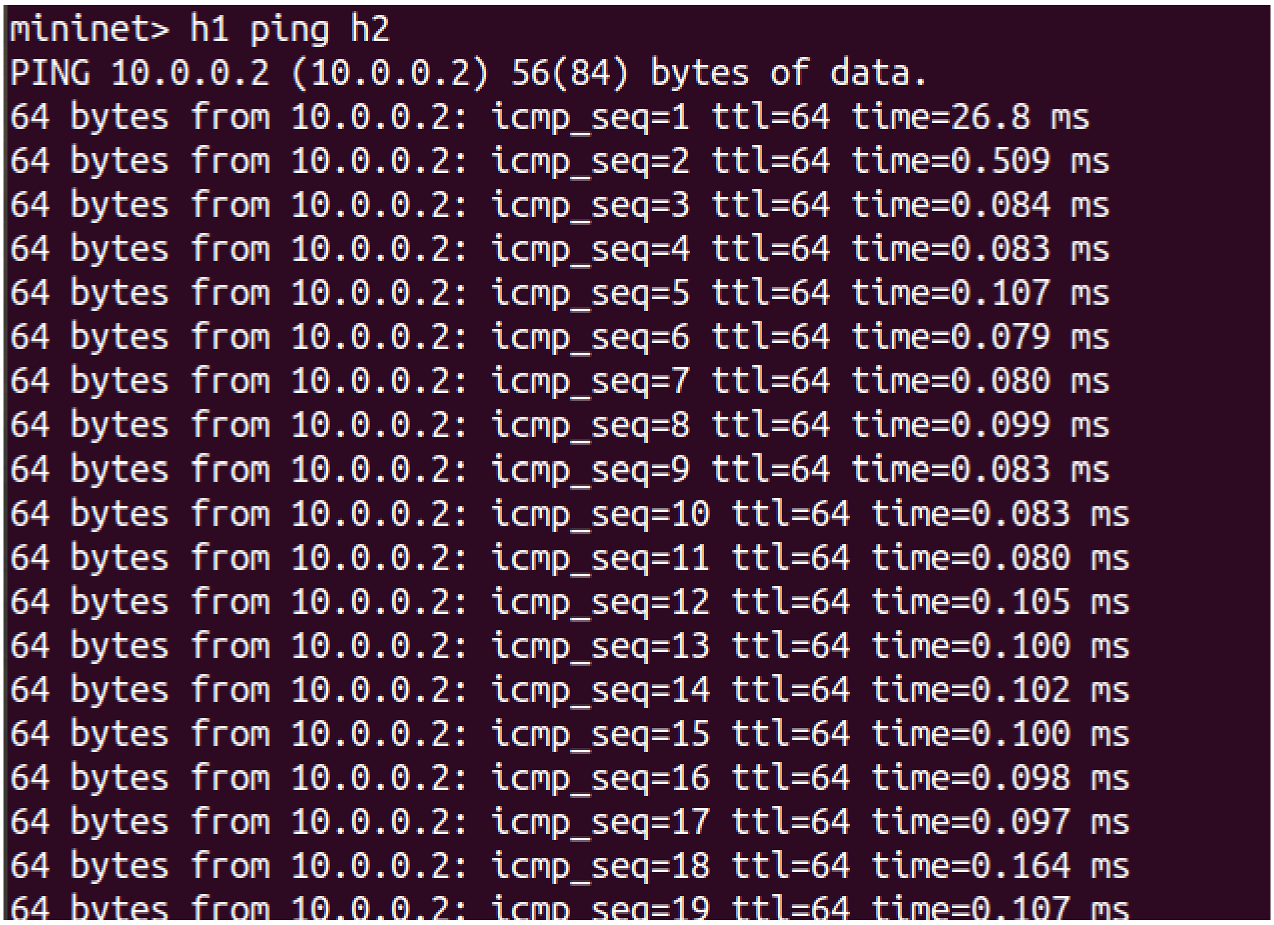
```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=8854>
<Host h2: h2-eth0:10.0.0.2 pid=8856>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=8861>
<Controller c0: 127.0.0.1:6653 pid=8847>
mininet> 
```

h1 ifconfig -a: This command will display the IP address, broadcast address and MAC address of the host *h1*. *T*he command **h2 ifconfig -a** are also as the command **h1 ifconfig -a.**



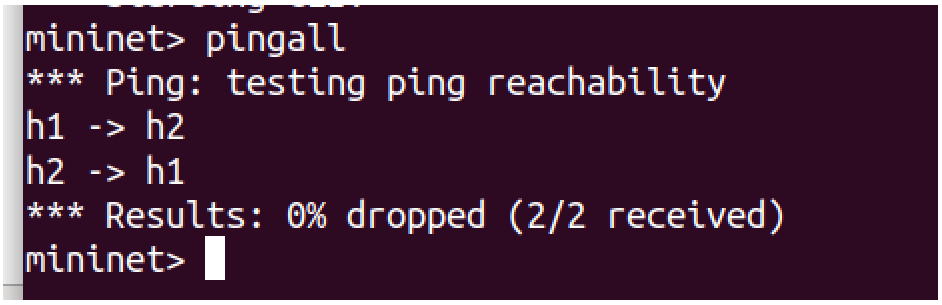
```
mininet> h1 ifconfig -a
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::f429:8aff:fe4d:e460  prefixlen 64  scopeid 0x20<link>
        ether f6:29:8a:4d:e4:60  txqueuelen 1000  (Ethernet)
        RX packets 61  bytes 7160 (7.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15  bytes 1146 (1.1 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>
```

ping command: We can test connectivity between the host by using ping command.



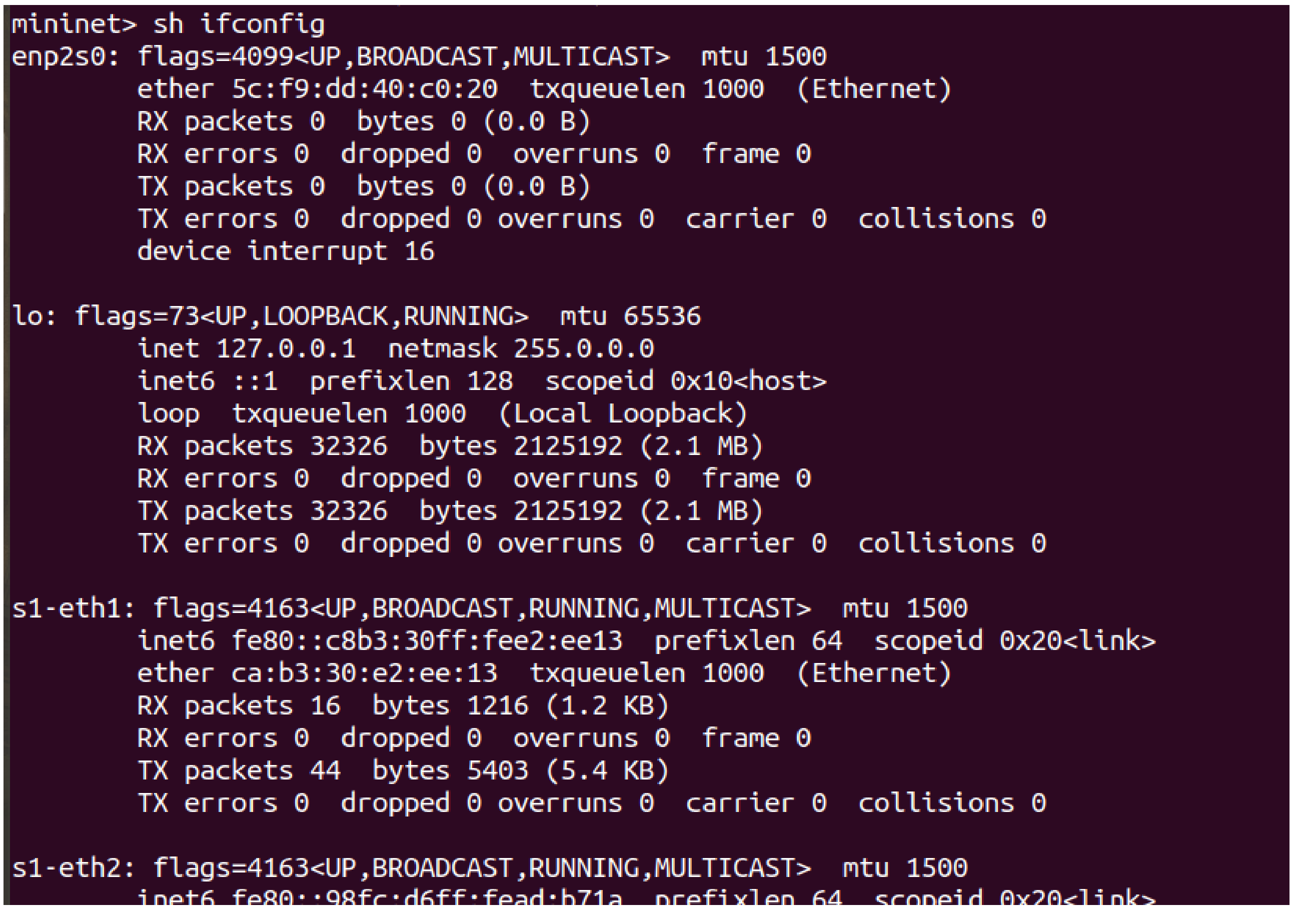
```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=26.8 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.509 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.084 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.083 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.107 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.079 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.080 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.099 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.083 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.083 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.080 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.105 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=0.102 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=0.098 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=0.097 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=0.164 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=0.107 ms
```

<u>Pingall command</u>: This command will make each host in the network ping every other host in the network. In the network that we have, *h1* will ping *h2*, and *h2* will ping *h1*.



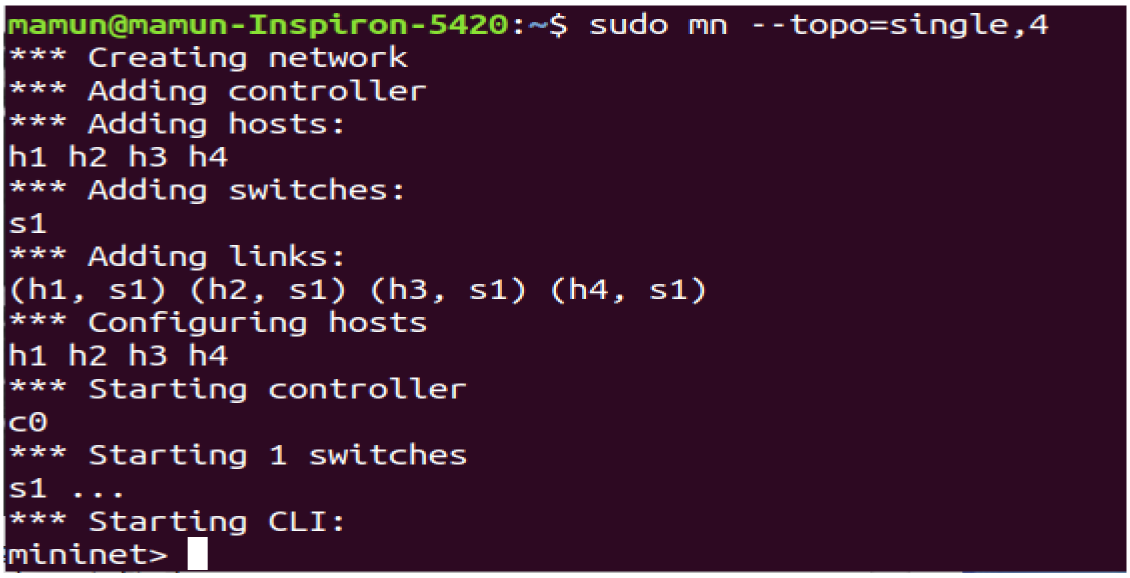
```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
mininet>
```

<u>Sh ifconfig command</u>: Check the interface on the host machine without exiting Mininet prompt



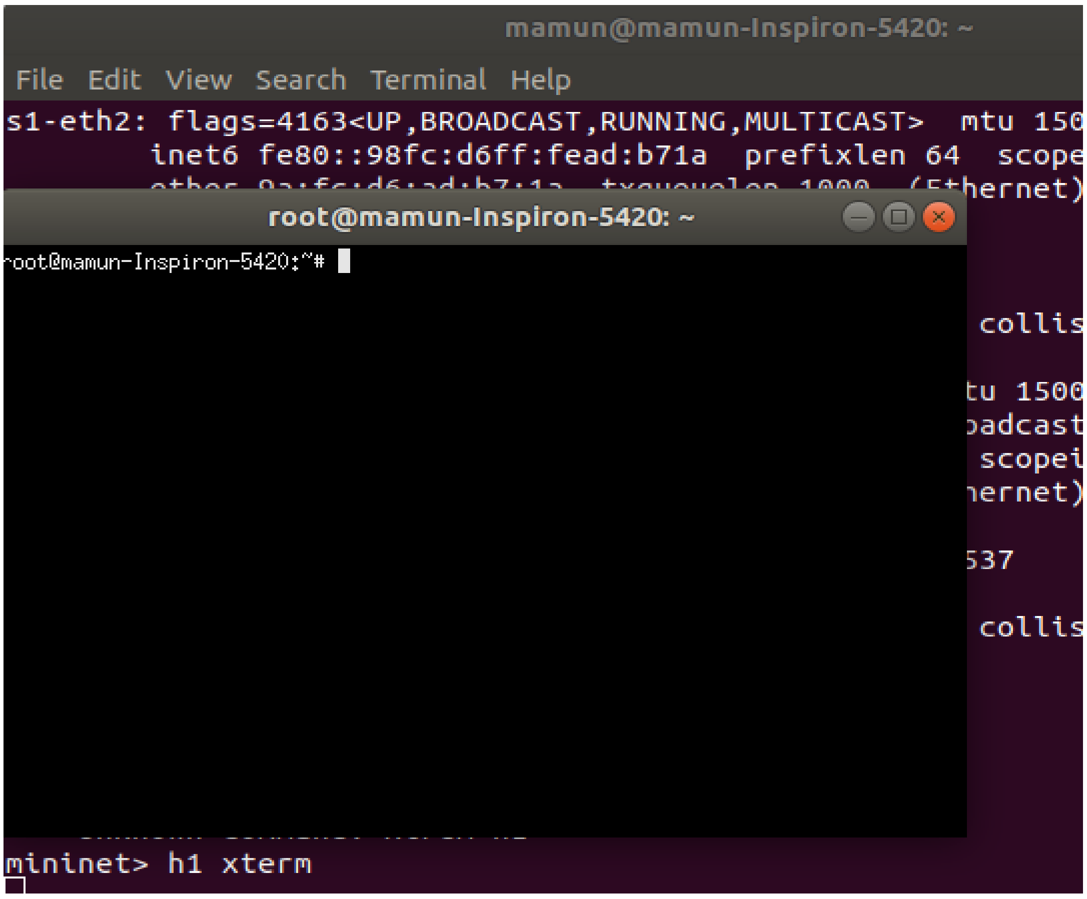
```
mininet> sh ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 5c:f9:dd:40:c0:20  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 16

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 32326  bytes 2125192 (2.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32326  bytes 2125192 (2.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::c8b3:30ff:fee2:ee13  prefixlen 64  scopeid 0x20<link>
        ether ca:b3:30:e2:ee:13  txqueuelen 1000  (Ethernet)
        RX packets 16  bytes 1216 (1.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 44  bytes 5403 (5.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::98fc:d6ff:fead:b71a  prefixlen 64  scopeid 0x20<link>
```

create topology : topo=single,4 that means controller , switch is one and host are 4 like h1,h2,h3,h4



```
mamun@mamun-Inspiron-5420:~$ sudo mn --topo=single,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>
```

xterm command: The command to open the xterm window is:



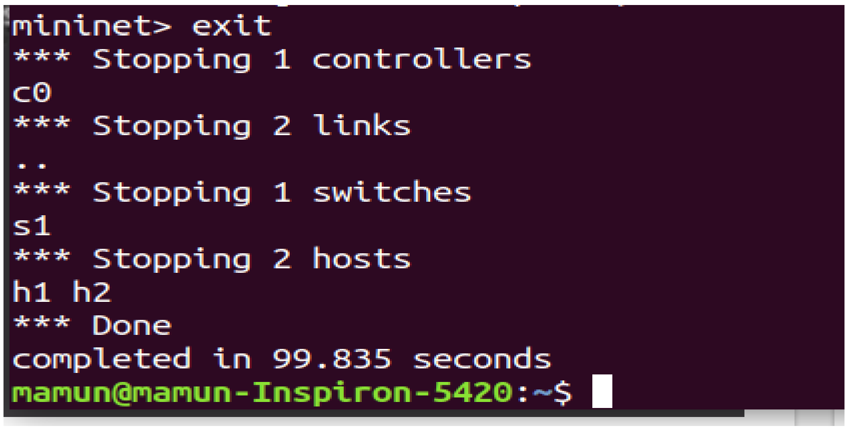
mamun@mamun-Inspiron-5420: ~
File Edit View Search Terminal Help
root@mamun-Inspiron-5420: ~

```
s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 150
        inet6 fe80::98fc:d6ff:fead:b71a  prefixlen 64  scope
```

```
root@mamun-Inspiron-5420:~#
```

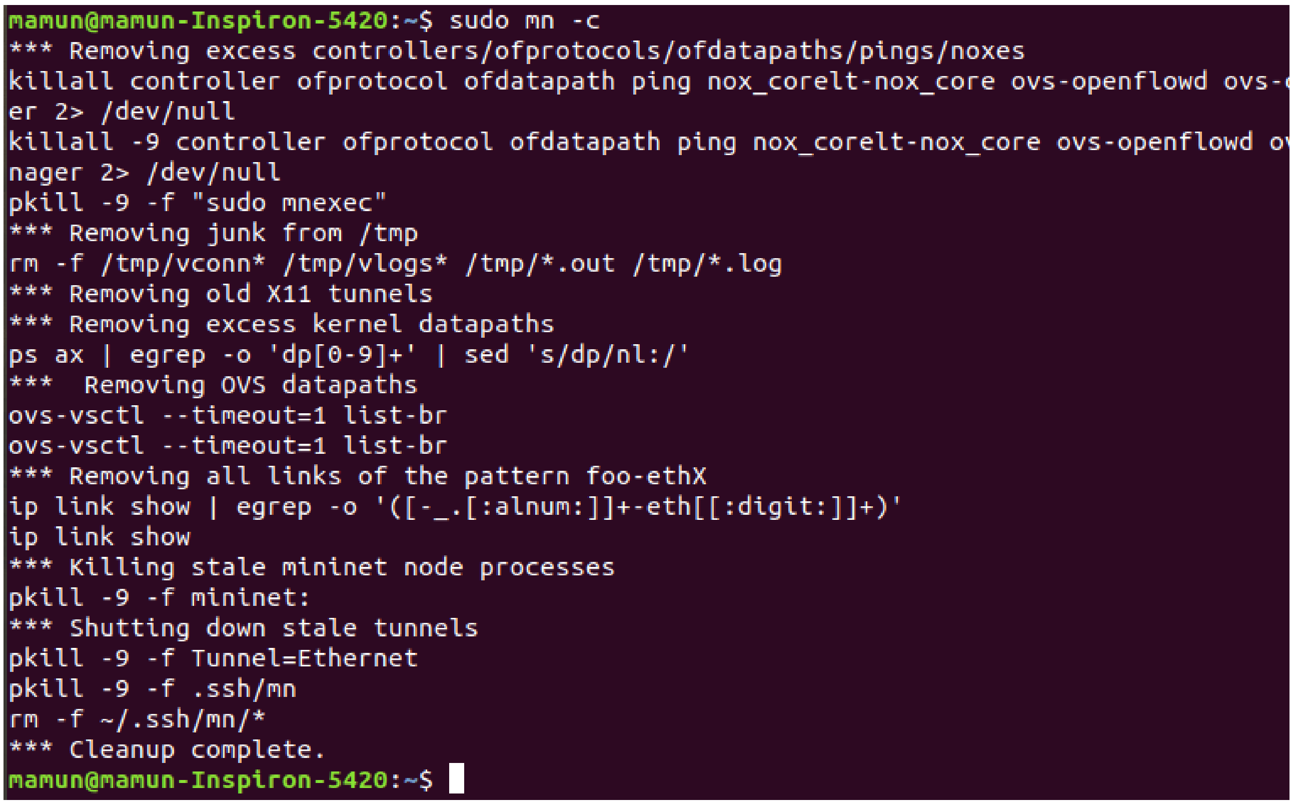
```
mininet> h1 xterm
```

exit command:



```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 99.835 seconds
mamun@mamun-Inspiron-5420:~$
```

cleanup command: If Mininet crashes for some reason, clean it up:



```
mamun@mamun-Inspiron-5420:~$ sudo mn -c
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_corelt-nox_core ovs-openflowd ovs-
er 2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_corelt-nox_core ovs-openflowd o
nager 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
***  Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '([-_.[:alnum:]]+-eth[[:digit:]]+)'
ip link show
*** Killing stale mininet node processes
pkill -9 -f mininet:
*** Shutting down stale tunnels
pkill -9 -f Tunnel=Ethernet
pkill -9 -f .ssh/mn
rm -f ~/.ssh/mn/*
*** Cleanup complete.
mamun@mamun-Inspiron-5420:~$
```

Discussion: This was an interesting lab. I can successfully understand the all things of this lab. I learned many things from this lab. This lab help me to understand the basic mininet command for Ubuntu terminal such as how to write commands and how to execute the commands etc. I also learn how to mininet work, advanced mininet and SDN controllers.