Mininet Introduction

Vipin Gupta

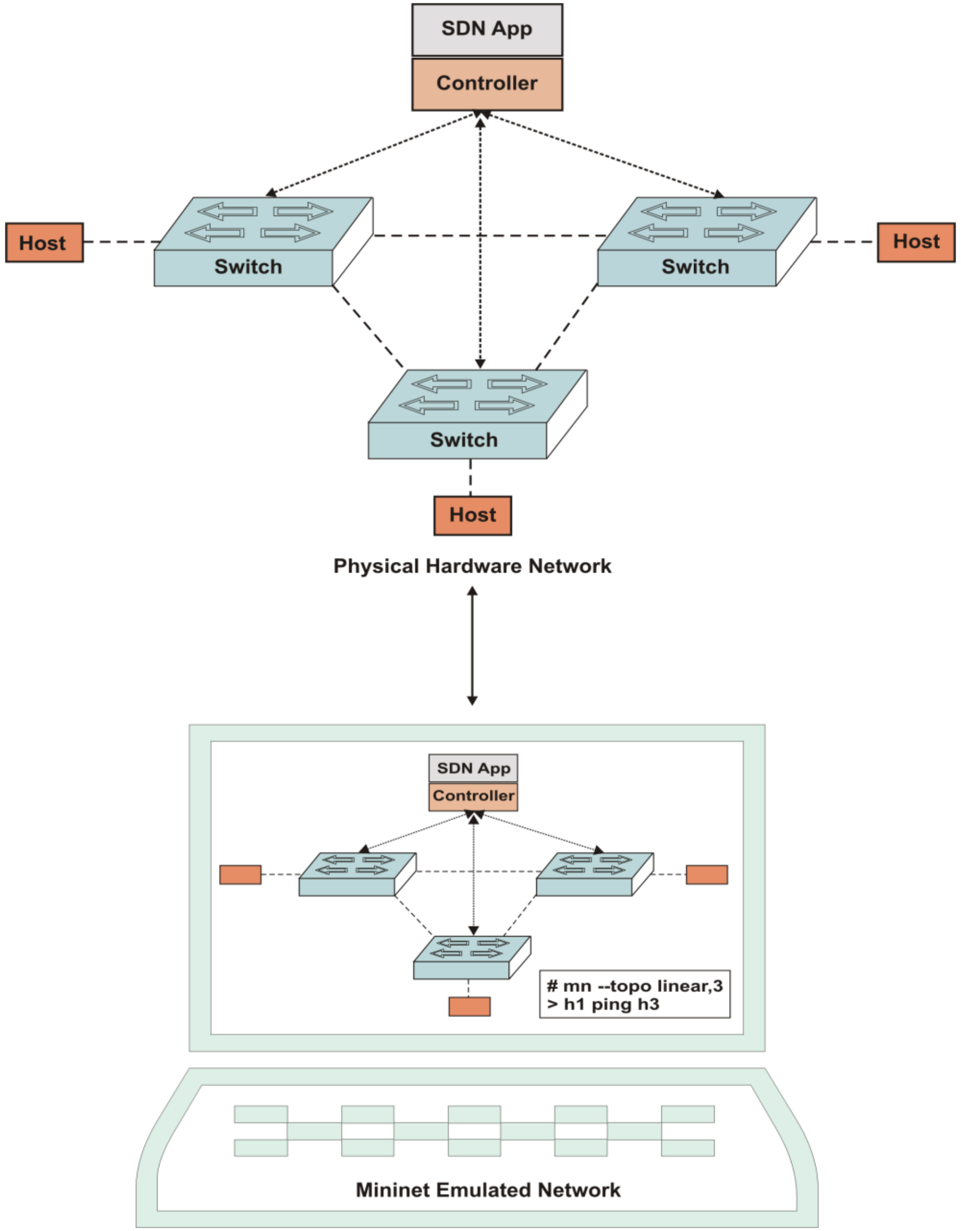
BE,RHCE,CEH,CCNA,MCSE,MCSA

vipin2411@gmail.com

Mobile: 93563-10379

www.linuxexpert.in

Introduction



```
# mn --topo linear,3
> h1 ping h3
```

To experiment with SDN, we need hosts, openflow switches, wires for connection between hosts & switches, wires for connecting switches & SDN controller. Mininet is a software program, which allows an entire network consisting of virtual hosts, controllers, switches, and links to be created and emulated on a single PC.

By using simple commands in mininet, we can create any type of topology. The minimal topology is 2 hosts, 1 switch & 1 controller. Large topologies could contain thousands of hosts, hundred of switches, links between them & controller. Network applications such as firewall, load balancer can be developed and tested on Mininet. The same application code can be moved to the actual production infrastructure.

Mininet vs Testbeds and Simulation Tools

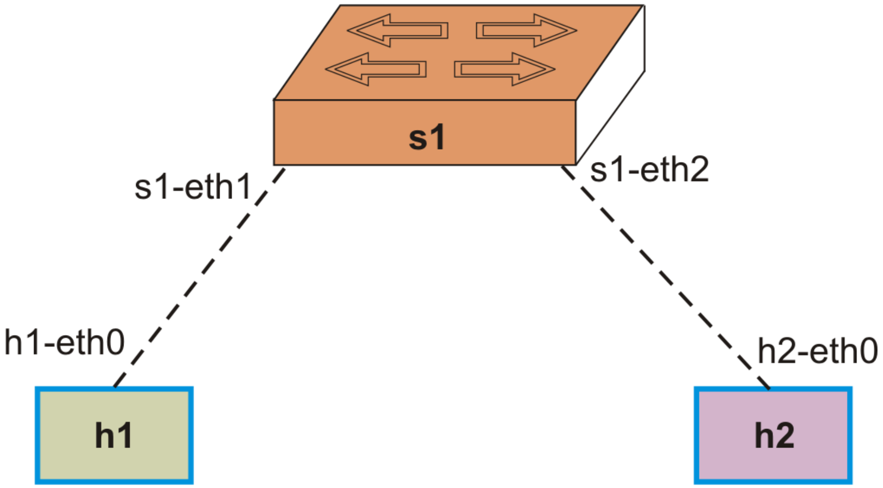
- Mininet is inexpensive, always available, quickly reconfigurable as compared to hardware testbeds such as GENI, VINI, FIRE, Emulab .

- As compared to simulators such as EstiNet, ns-3, Mininet runs real, unmodified code.

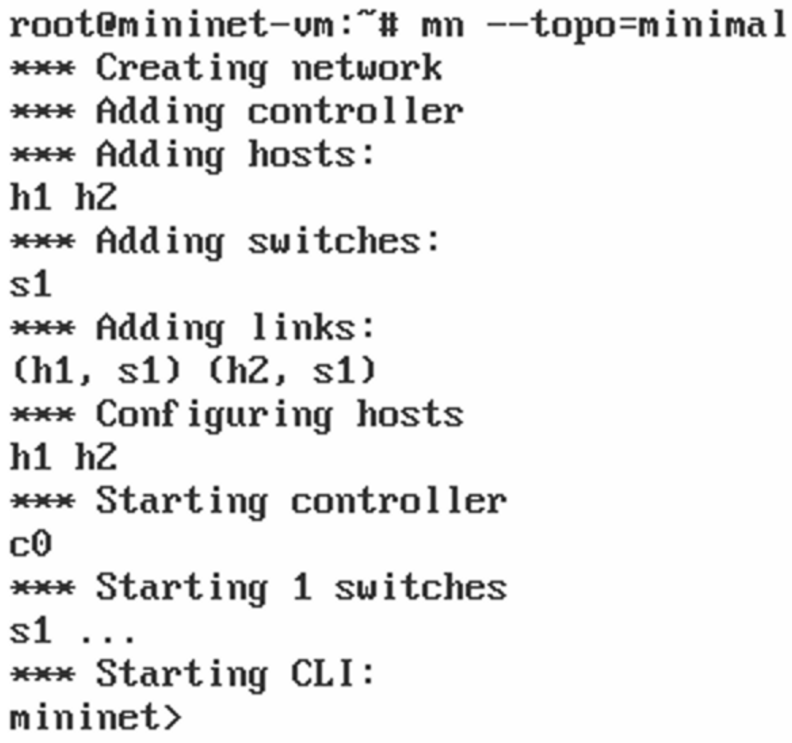
Different Mininet Topologies

- Minimal
- Single
- Reversed
- Linear
- Tree
- Custom

Minimal Topology



It consists of 2 hosts & 1 openflow switch

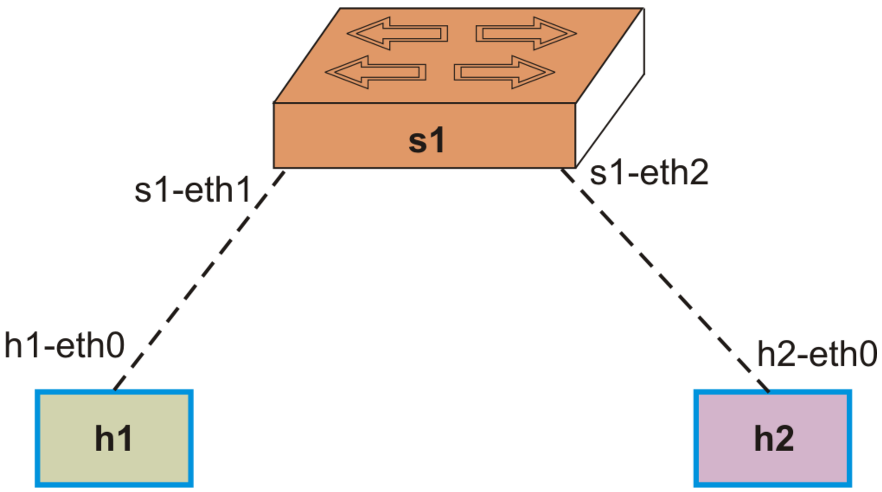


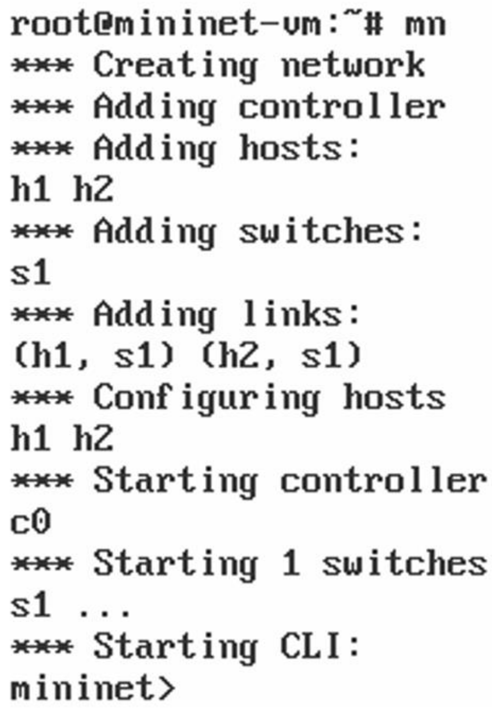
```
root@mininet-vm:~# mn --topo=minimal
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

```
# mn --topo=minimal
or
# mn
```

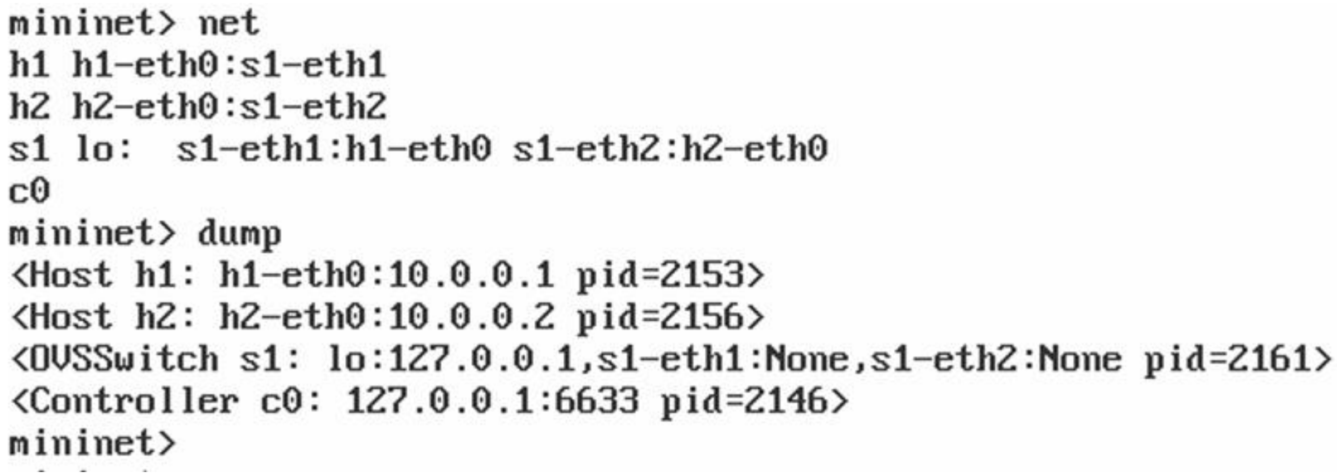
switch/host	interfaces	ip address	mac address
s1	s1-eth1,s1-eth2	---	random
h1	h1-eth0	10.0.0.1	random
h2	h2-eth0	10.0.0.2	random

Minimal Topology





```
root@mininet-vm:~# mn
*** 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>
```



```
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
<Host h1: h1-eth0:10.0.0.1 pid=2153>
<Host h2: h2-eth0:10.0.0.2 pid=2156>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2161>
<Controller c0: 127.0.0.1:6633 pid=2146>
mininet>
```

Naming Rules

Hosts: h1 ... hn
Switches: s1 ... sn

Naming of hosts, switches and interfaces

- Hosts are named *h1 … hn*

- Switches are named *s1 … sn*.

- Host Interfaces names are created by using hostname prefix followed by ethernet name starting with zero.

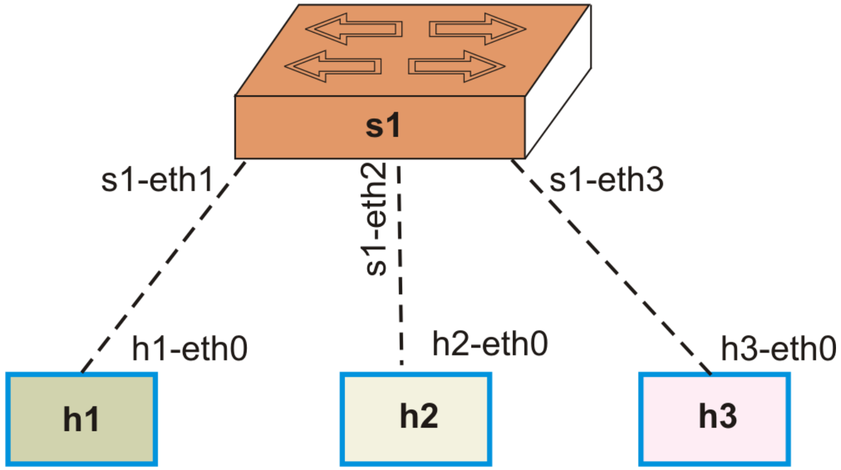
- host h1 with 1 interface will have the name “h1-eth0”. if host *h1* has got 3 interfaces. First interface will be called *“h1-eth0“, 2nd “h1-eth1“, 3rd one “h1-eth2“* & so on. In hosts numbering begins with 0.

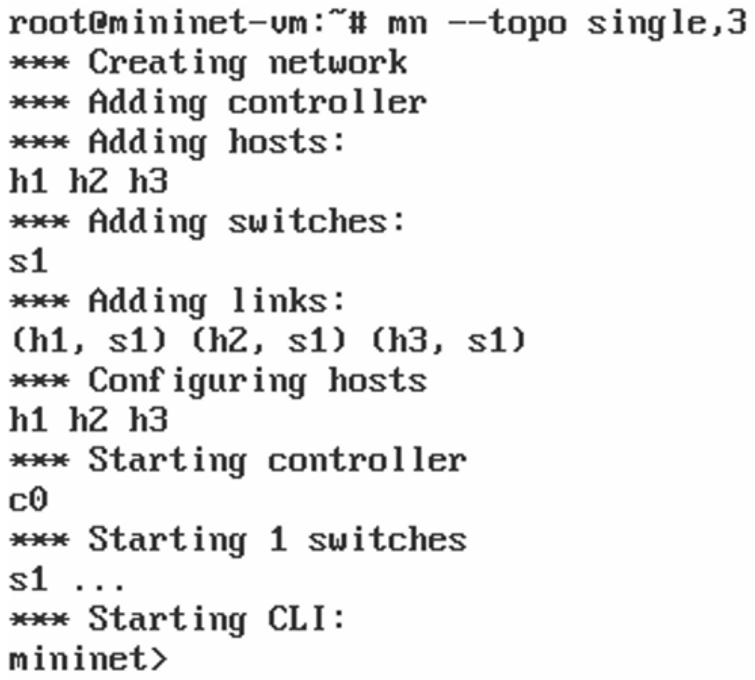
- switch Interfaces names are created by using switch name prefix followed by ethernet name starting with 1.

- In switches port numbering starts from 1. ie switch s1 with four interface will have interface names s1-eth1, s1-eth2, s1-eth3 & s1-eth4. So 3rd port in switch *“s2“* will be called *“s2-eth3“*.

Single Topology

Single Topology





```
root@mininet-vm:~# mn --topo single,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

The syntax for single topology is
# mn --topo single, n

It consists of n hosts & 1 openflow switch

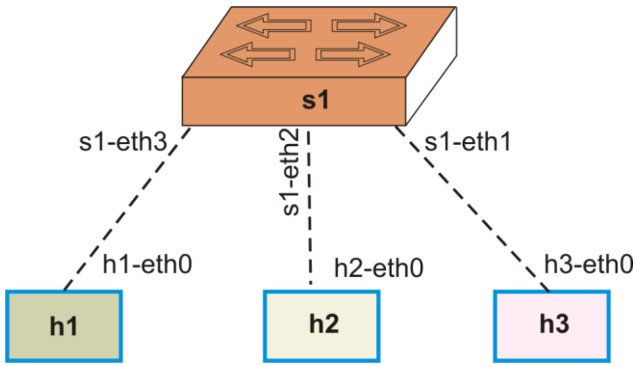
# mn --topo single, 3

will create 1 switch and 3 hosts

switch/host	interfaces	ip address
s1	s1-eth1,s1-eth2,s1-eth3	---
h1	h1-eth0	10.0.0.1
h2	h2-eth0	10.0.0.2
h3	h3-eth0	10.0.0.3

Reversed Topology

Reversed Topology

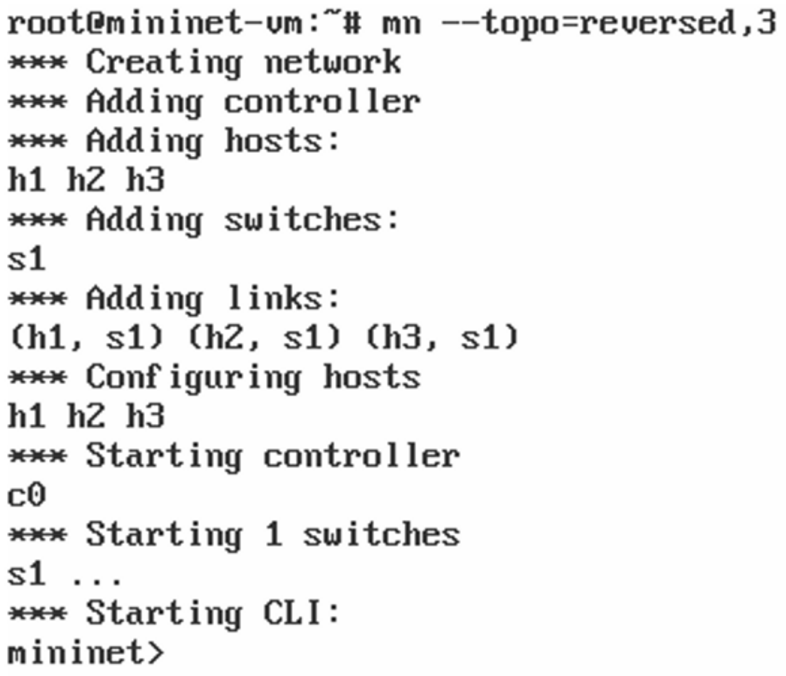


The syntax for reversed topology is
# mn --topo reversed, n

It consists of n hosts & 1 openflow switch. It is similar to single topology but connections are in reversed order

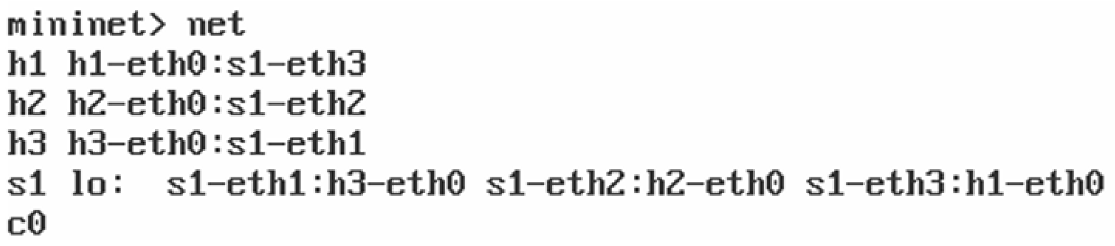
mn --topo reversed, 3

will create 1 switch and 3 hosts



```
root@mininet-vm:~# mn --topo=reversed,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

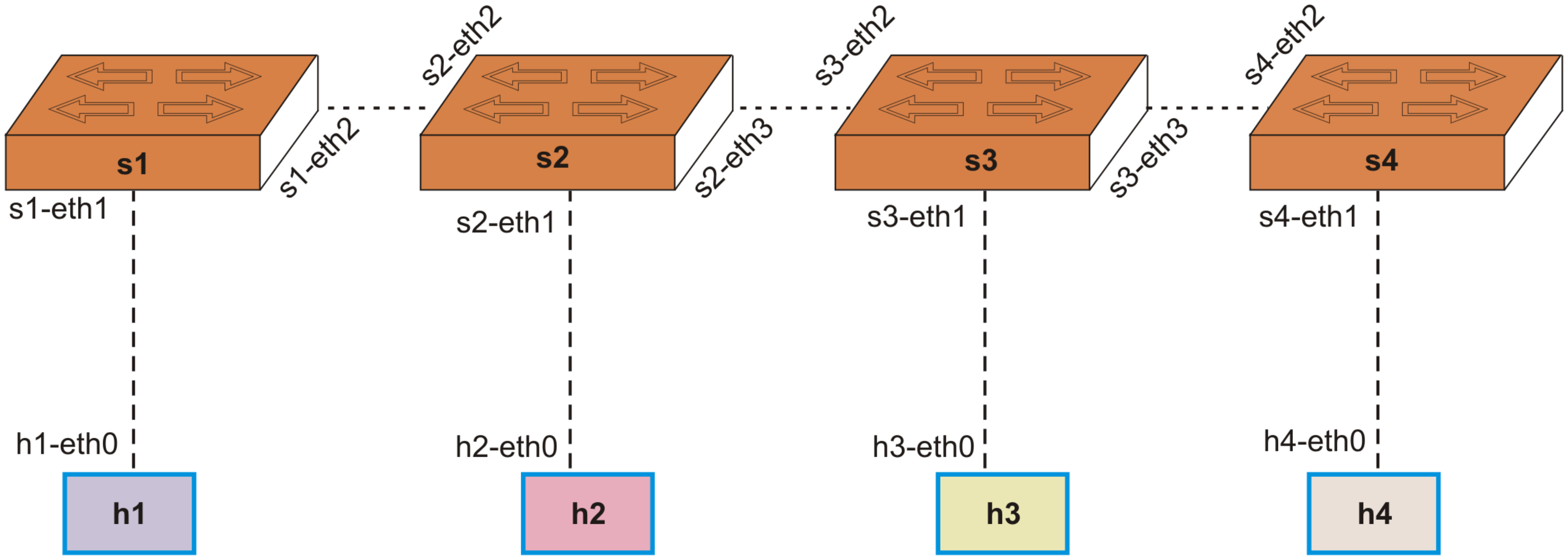
switch/host	interfaces	ip address
s1	s1-eth3,s1-eth2,s1-eth1	---
h1	h1-eth0	10.0.0.1
h2	h2-eth0	10.0.0.2
h3	h3-eth0	10.0.0.3



```
mininet> net
h1 h1-eth0:s1-eth3
h2 h2-eth0:s1-eth2
h3 h3-eth0:s1-eth1
s1 lo:  s1-eth1:h3-eth0 s1-eth2:h2-eth0 s1-eth3:h1-eth0
c0
```

Linear Topology

Linear Topology

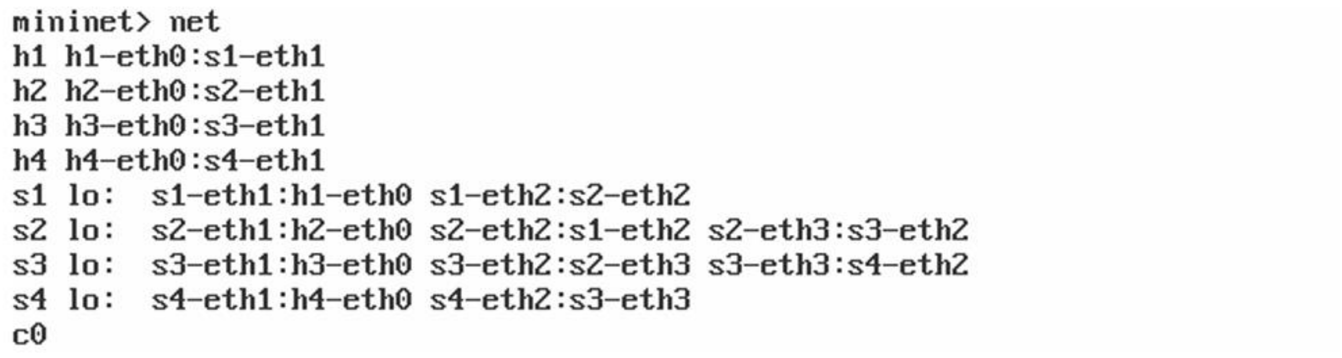


The syntax for linear topology is
# mn --topo linear, n

Linear topology consists of n number of switches & n number of hosts

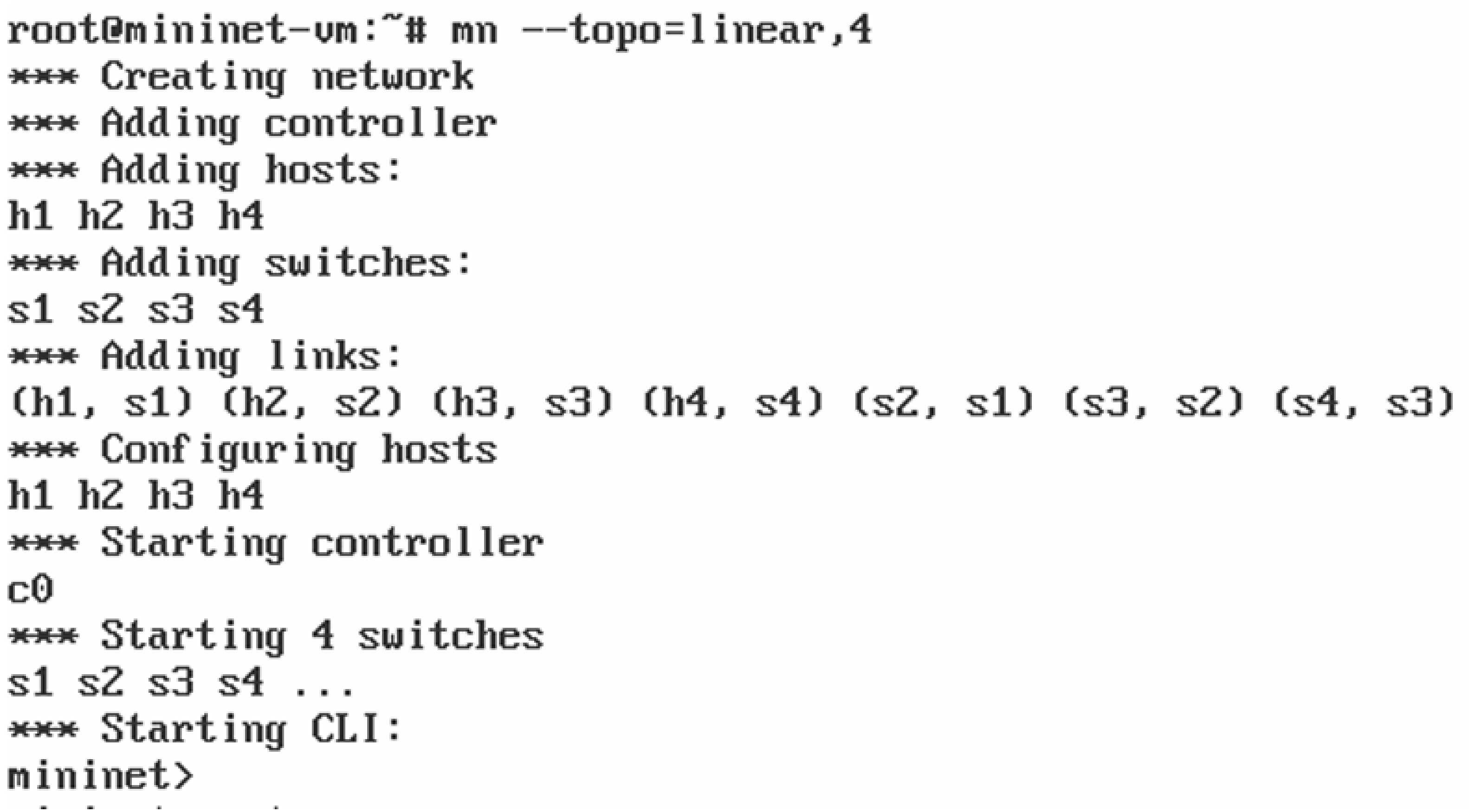
# mn --topo linear, 4

will create 4 switches and 4 hosts



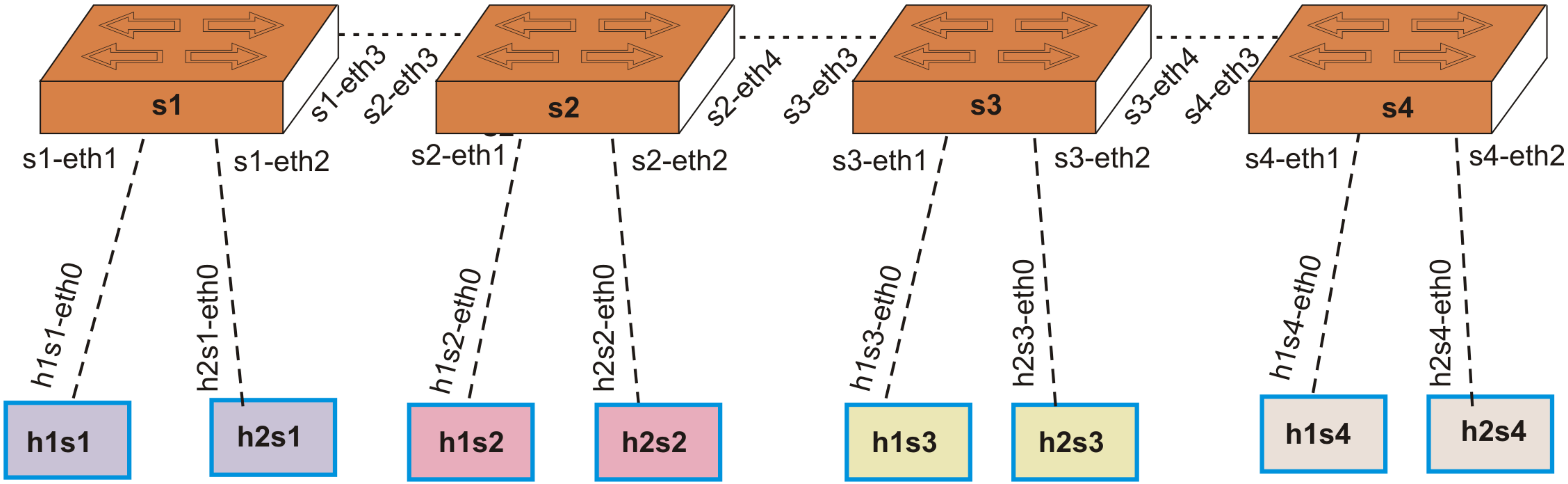
```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
h4 h4-eth0:s4-eth1
s1 lo:  s1-eth1:h1-eth0 s1-eth2:s2-eth2
s2 lo:  s2-eth1:h2-eth0 s2-eth2:s1-eth2 s2-eth3:s3-eth2
s3 lo:  s3-eth1:h3-eth0 s3-eth2:s2-eth3 s3-eth3:s4-eth2
s4 lo:  s4-eth1:h4-eth0 s4-eth2:s3-eth3
c0
```

Linear Topology



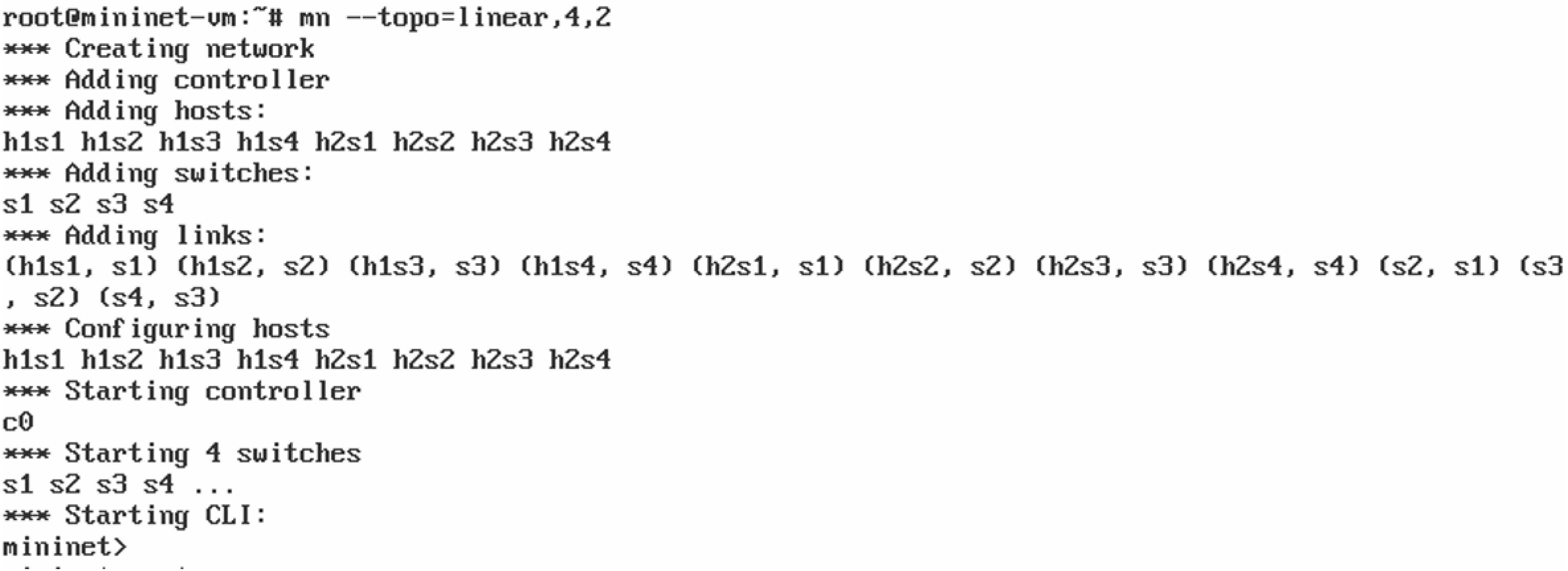
```
root@mininet-vm:~# mn --topo=linear,4
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (s2, s1) (s3, s2) (s4, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

Linear Topology



What will be the command if we want to attach 2 hosts with each switch ?

Linear Topology



```
root@mininet-vm:~# mn --topo=linear,4,2
*** Creating network
*** Adding controller
*** Adding hosts:
h1s1 h1s2 h1s3 h1s4 h2s1 h2s2 h2s3 h2s4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1s1, s1) (h1s2, s2) (h1s3, s3) (h1s4, s4) (h2s1, s1) (h2s2, s2) (h2s3, s3) (h2s4, s4) (s2, s1) (s3
, s2) (s4, s3)
*** Configuring hosts
h1s1 h1s2 h1s3 h1s4 h2s1 h2s2 h2s3 h2s4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

mn --topo linear, 4,2

Tree Topology

Tree Topology

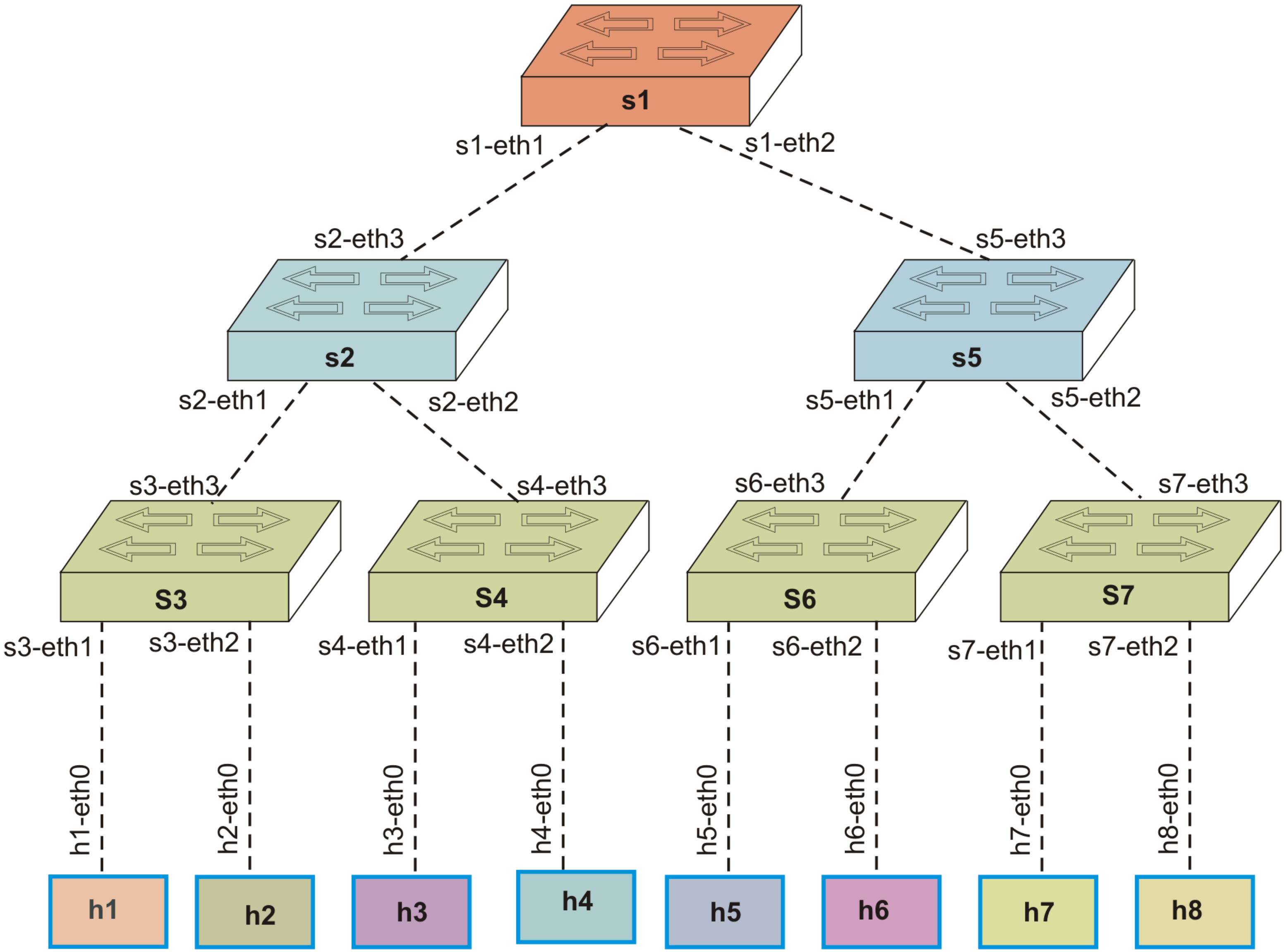
This topology contains n levels and default 2 hosts are attached

The syntax for tree topology is

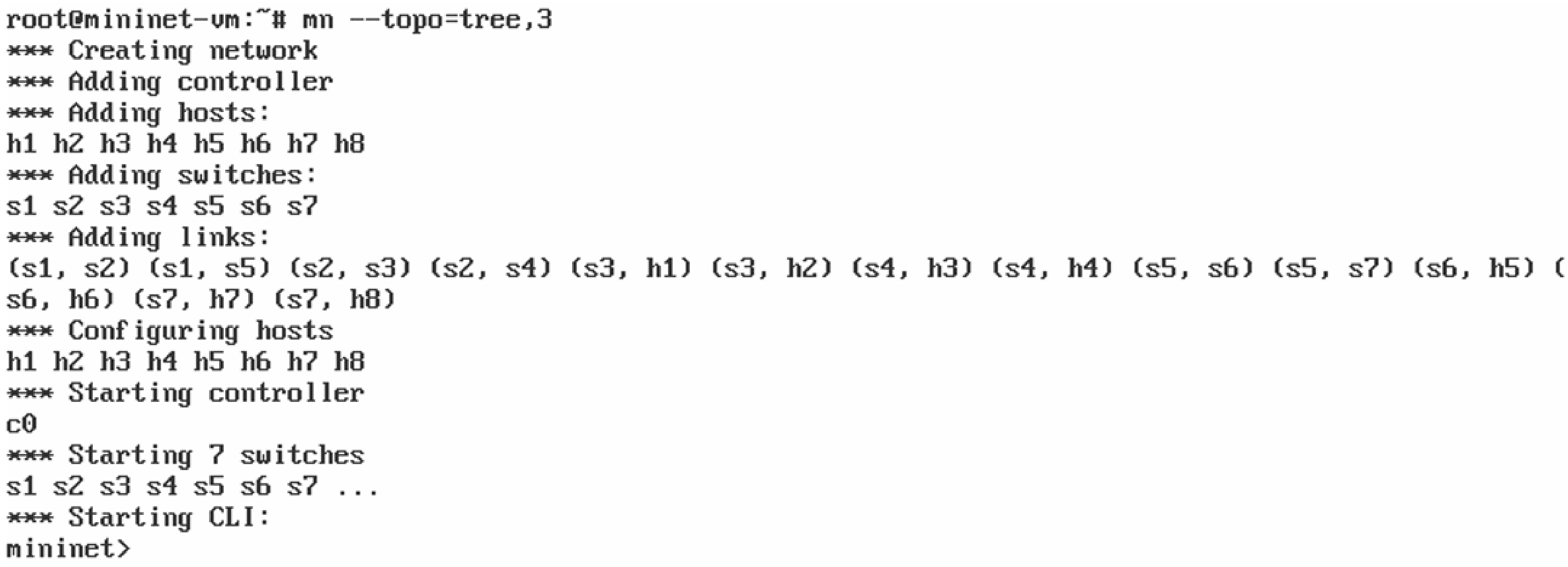
```
# mn --topo tree, n
```

```
# mn --topo tree, 3
```

will create a topology 3 levels deep. There will be two hosts attached with each access switch.



Tree Topology



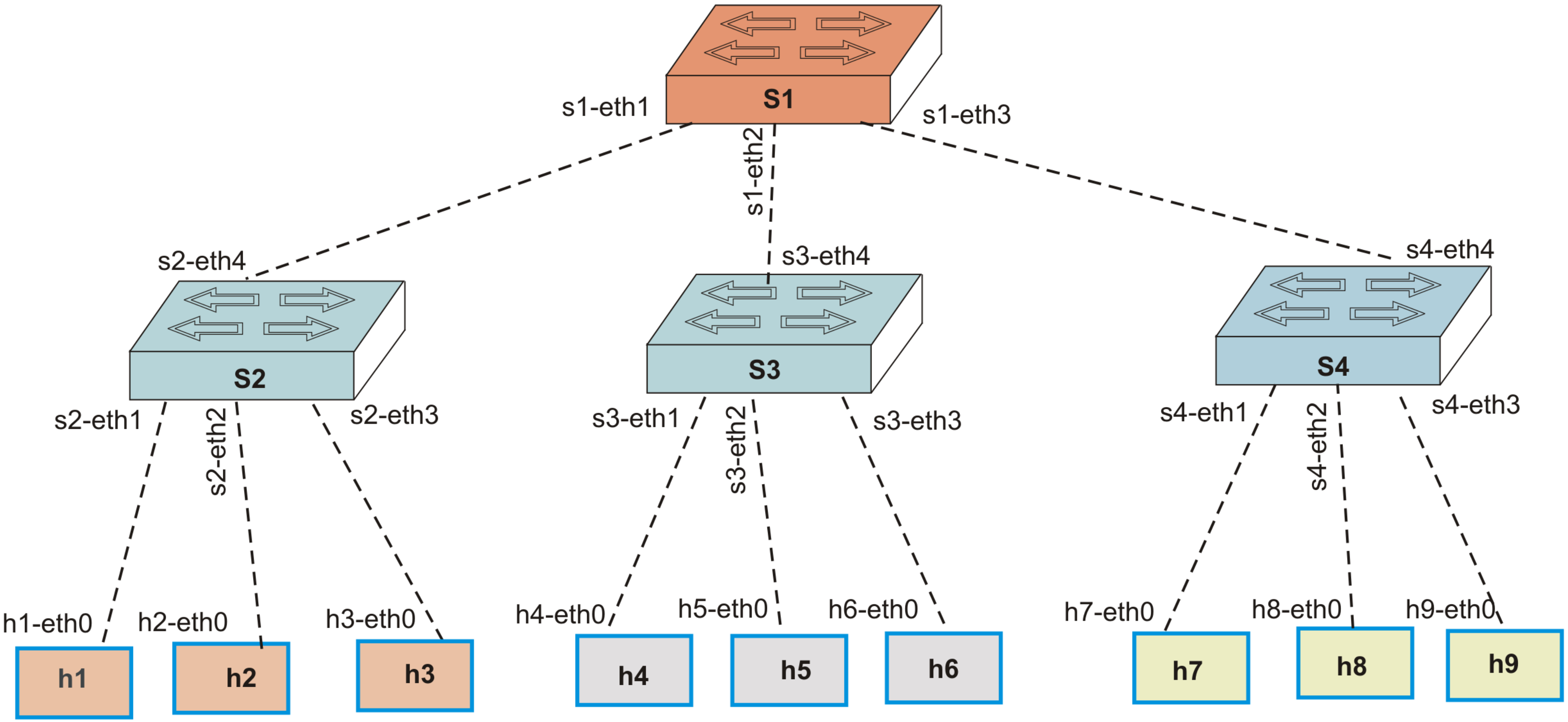
```
root@mininet-vm:~# mn --topo=tree,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7
*** Adding links:
(s1, s2) (s1, s5) (s2, s3) (s2, s4) (s3, h1) (s3, h2) (s4, h3) (s4, h4) (s5, s6) (s5, s7) (s6, h5) (
s6, h6) (s7, h7) (s7, h8)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8
*** Starting controller
c0
*** Starting 7 switches
s1 s2 s3 s4 s5 s6 s7 ...
*** Starting CLI:
mininet>
```

Tree Topology

what will the following command will do

```
# mn –topo tree,2,3
```

Tree Topology

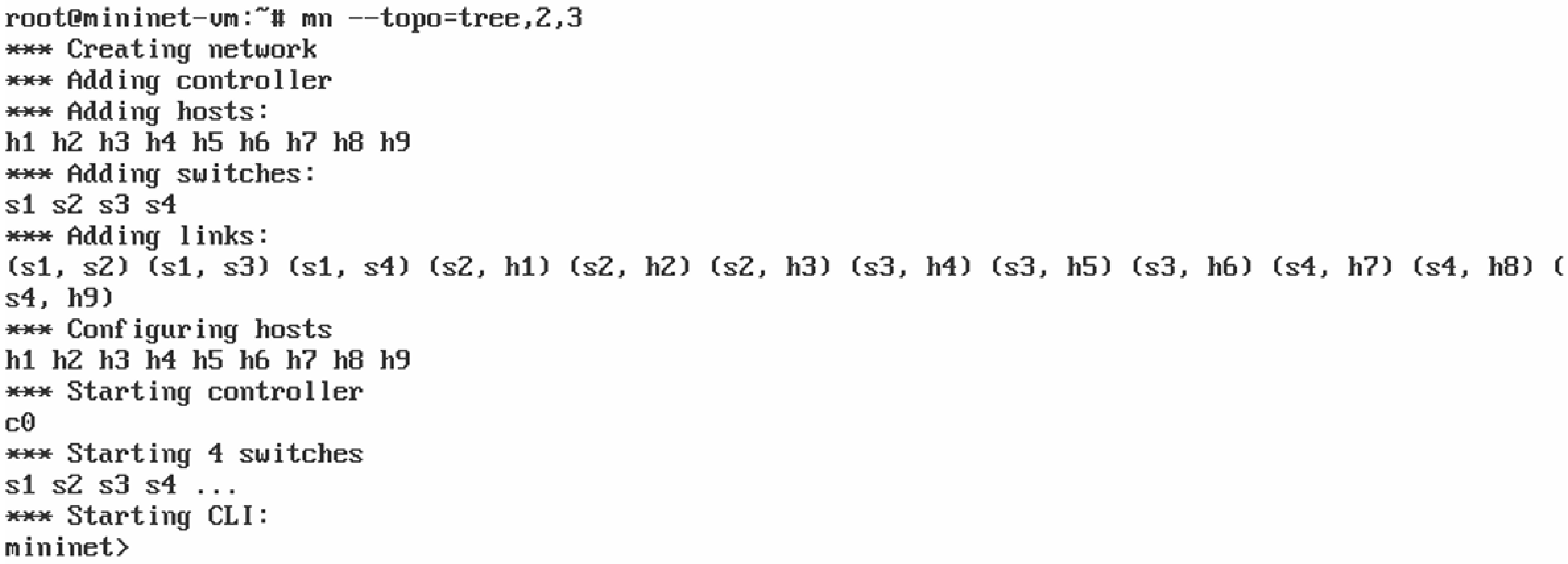


what will the following command will do

# mn --topo,2,3

will create a topology 2 levels deep. There will be three hosts attached with each access

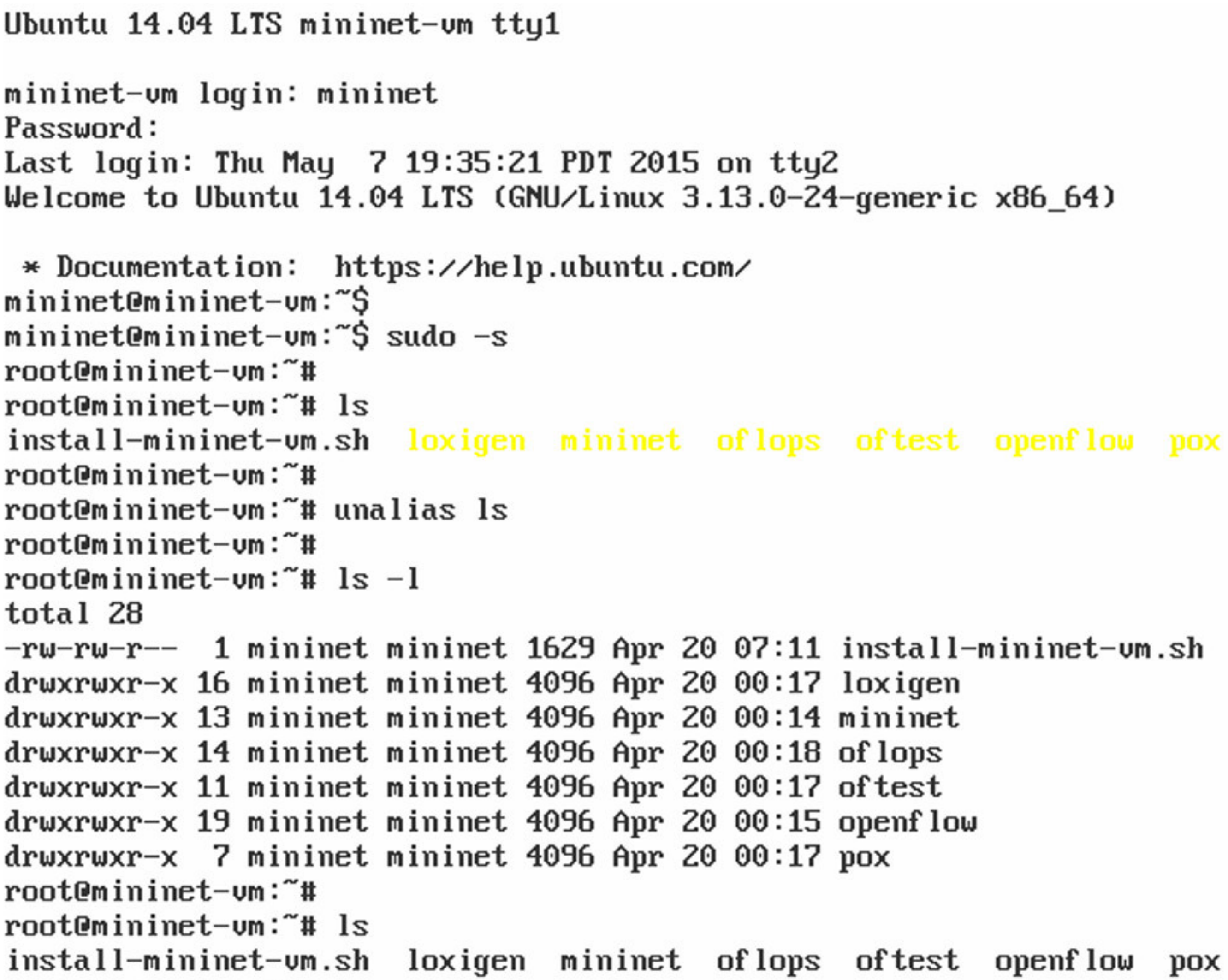
Tree Topology



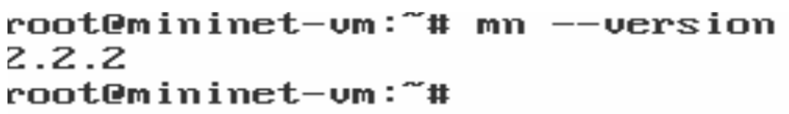
```
root@mininet-vm:~# mn --topo=tree,2,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s2, h1) (s2, h2) (s2, h3) (s3, h4) (s3, h5) (s3, h6) (s4, h7) (s4, h8) (
s4, h9)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
*** Starting CLI:
mininet>
```

Mininet Commands & Options

Mininet Commands



```
Ubuntu 14.04 LTS mininet-vm tty1

mininet-vm login: mininet
Password:
Last login: Thu May  7 19:35:21 PDT 2015 on tty2
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
mininet@mininet-vm:~$
mininet@mininet-vm:~$ sudo -s
root@mininet-vm:~#
root@mininet-vm:~# ls
install-mininet-vm.sh  loxigen  mininet  oflops  oftest  openflow  pox
root@mininet-vm:~#
root@mininet-vm:~# unalias ls
root@mininet-vm:~#
root@mininet-vm:~# ls -l
total 28
-rw-rw-r--  1 mininet mininet 1629 Apr 20 07:11 install-mininet-vm.sh
drwxrwxr-x 16 mininet mininet 4096 Apr 20 00:17 loxigen
drwxrwxr-x 13 mininet mininet 4096 Apr 20 00:14 mininet
drwxrwxr-x 14 mininet mininet 4096 Apr 20 00:18 oflops
drwxrwxr-x 11 mininet mininet 4096 Apr 20 00:17 oftest
drwxrwxr-x 19 mininet mininet 4096 Apr 20 00:15 openflow
drwxrwxr-x  7 mininet mininet 4096 Apr 20 00:17 pox
root@mininet-vm:~#
root@mininet-vm:~# ls
install-mininet-vm.sh  loxigen  mininet  oflops  oftest  openflow  pox
```



```
root@mininet-vm:~# mn --version
2.2.2
root@mininet-vm:~#
```

mn --version

(It will display the version of mininet which is "2.2.2" in our case)

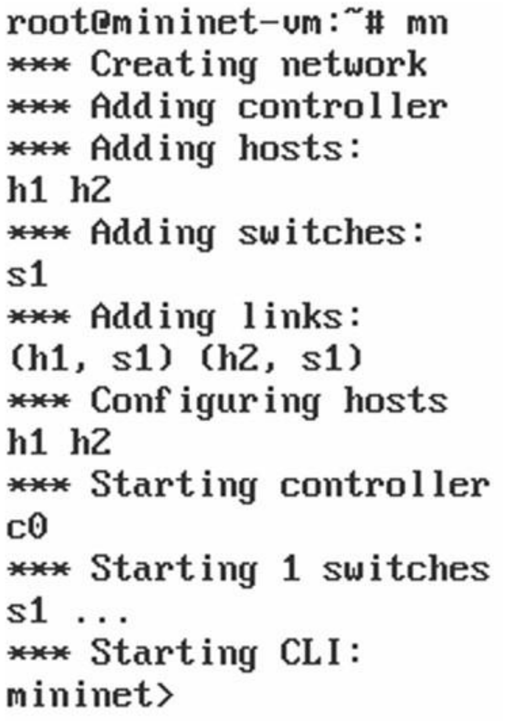
# mn -h

(will display the help for "mn" command)

Run the "ls" command for viewing the directories available. Sometimes the colored output is not comfortable for our eyes, so you stop colored display of the output by using "unalias ls" command.

Now run "ls –l" to view information about files & directories in greater details. There are many directories available out of which 2 are very important namely "mininet" & "pox"

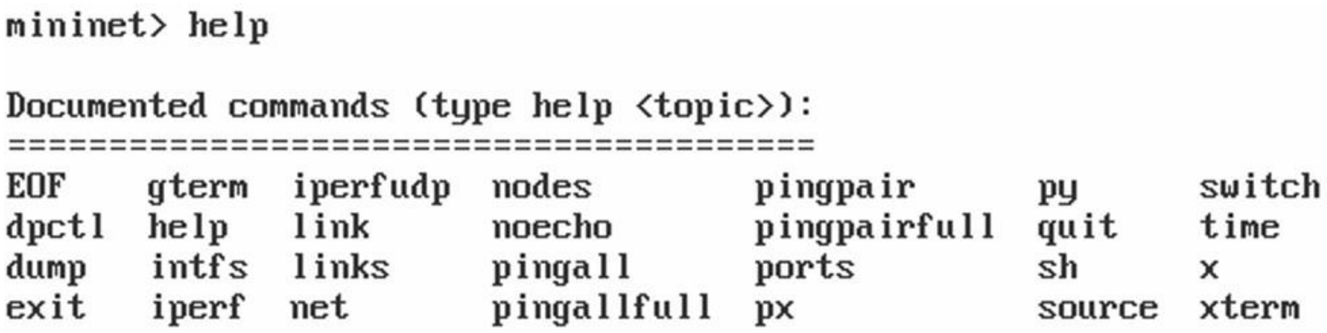
Mininet Commands



```
root@mininet-vm:~# mn
*** 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>
```

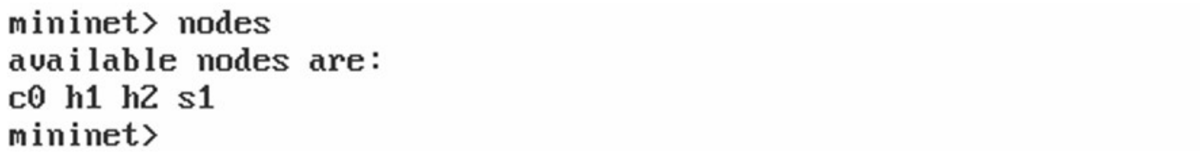
mn

(It will create a minimal topology consisting of 2 hosts h1 & h2, 1 switch s1 & 1 controller c0. It will takes you to “mininet>” prompt, where we can type various commands. Type “help” for information about various commands & syntax for running commands)

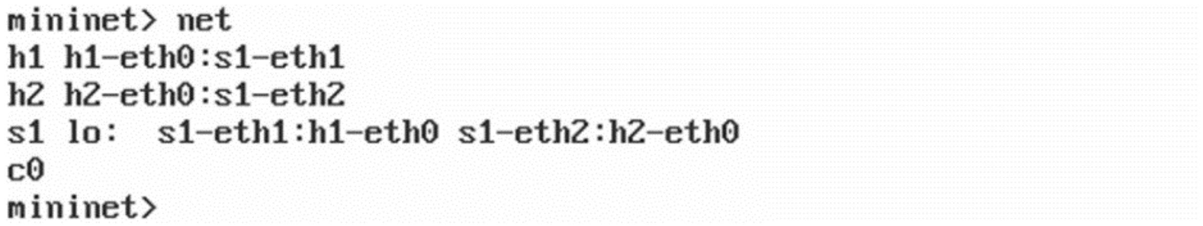


```
mininet> help

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
```

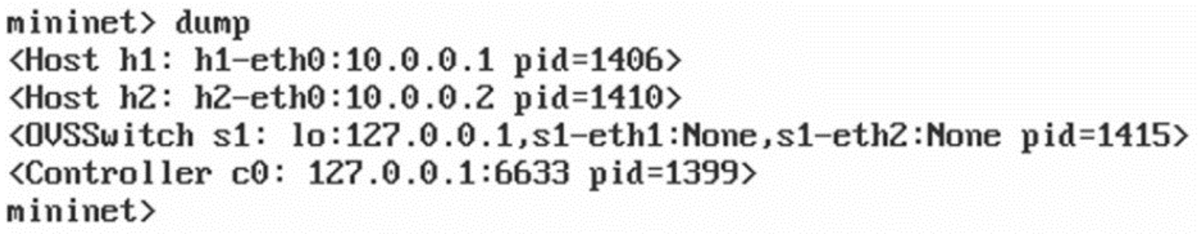
Mininet Commands



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



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



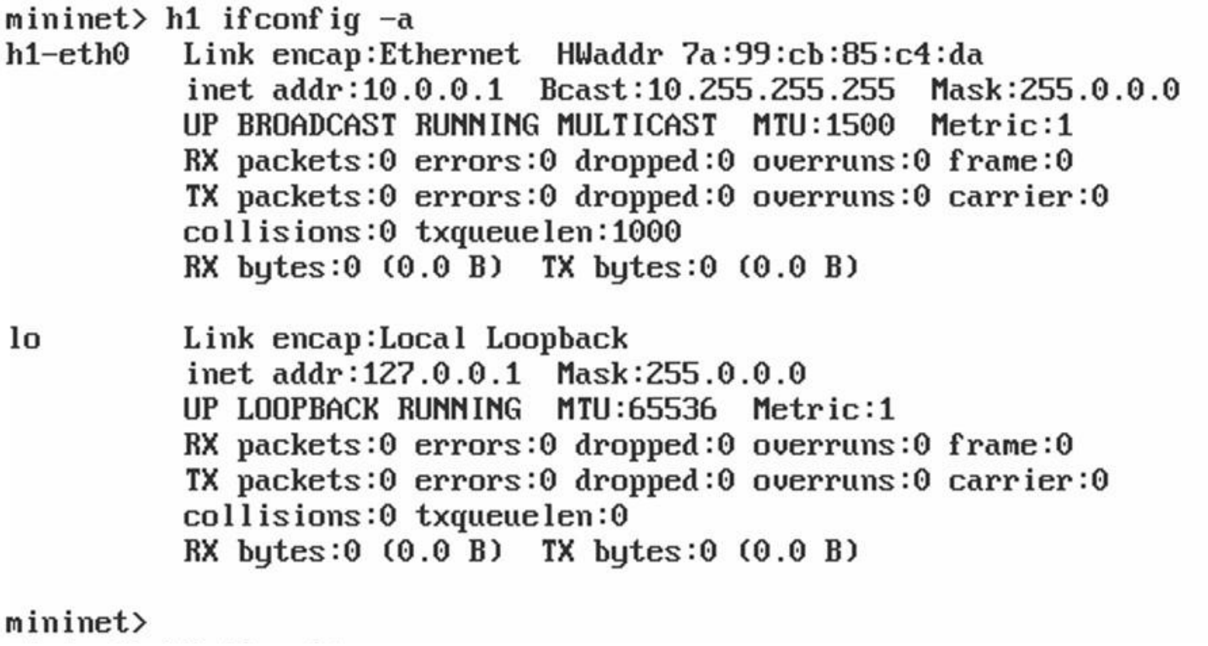
```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=1406>
<Host h2: h2-eth0:10.0.0.2 pid=1410>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=1415>
<Controller c0: 127.0.0.1:6633 pid=1399>
mininet>
```

mininet> nodes
(Display information about nodes. Hosts, Switches & Controllers are all nodes)

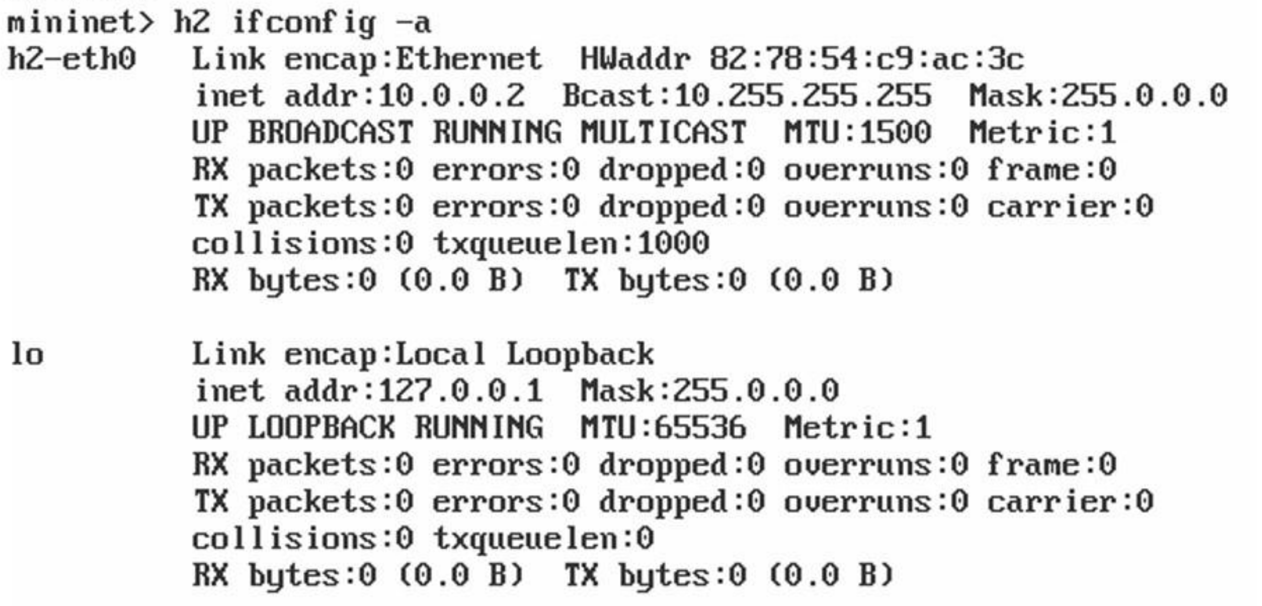
mininet> net
(Display information about links between hosts, switches. Here it is showing that network interface “h1-eth0” on host h1 is connected with “s1-eth1” interface of switch s1)

mininet> dump
(dump detailed information about all nodes)

Mininet Commands



```
mininet> h1 ifconfig -a
h1-eth0   Link encap:Ethernet  HWaddr 7a:99:cb:85:c4:da
          inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet>
```

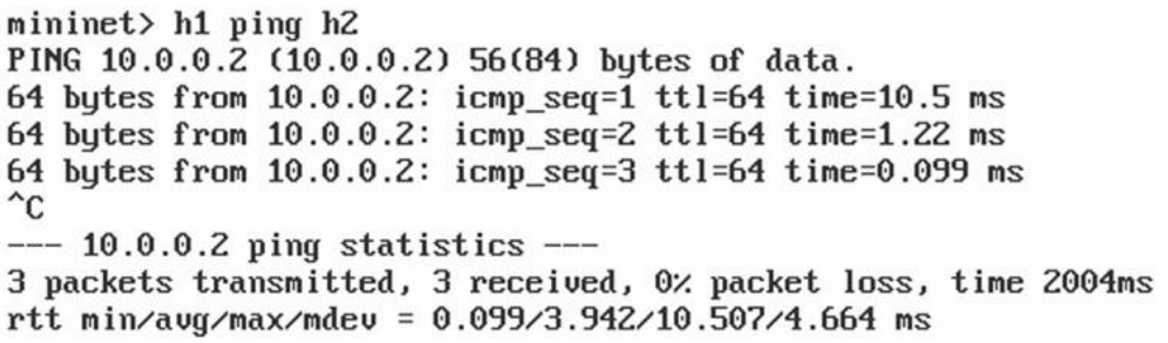


```
mininet> h2 ifconfig -a
h2-eth0   Link encap:Ethernet  HWaddr 82:78:54:c9:ac:3c
          inet addr:10.0.0.2  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

mininet> h1 ifconfig –a

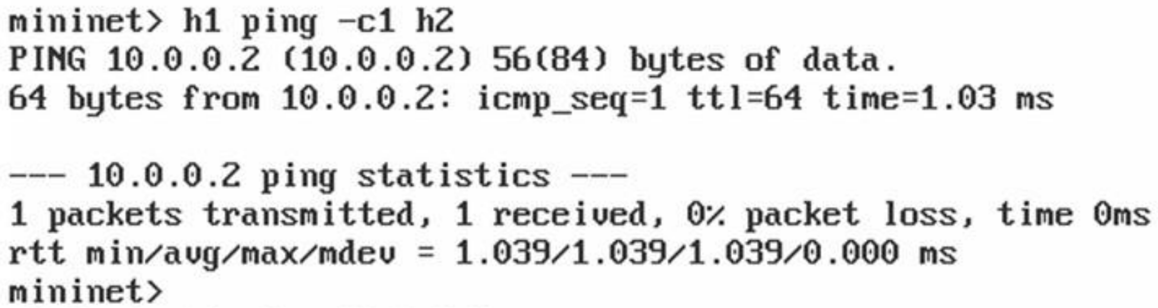
mininet> h2 ifconfig -a
(Show ip addresses on hosts h1 & h2)

mininet> h1 ping h2
(From host h1 ping to host h2)



```
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=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.22 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.099 ms
^C
--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 0.099/3.942/10.507/4.664 ms
```

Mininet Commands



```
mininet> h1 ping -c1 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=1.03 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.039/1.039/1.039/0.000 ms
mininet>
```

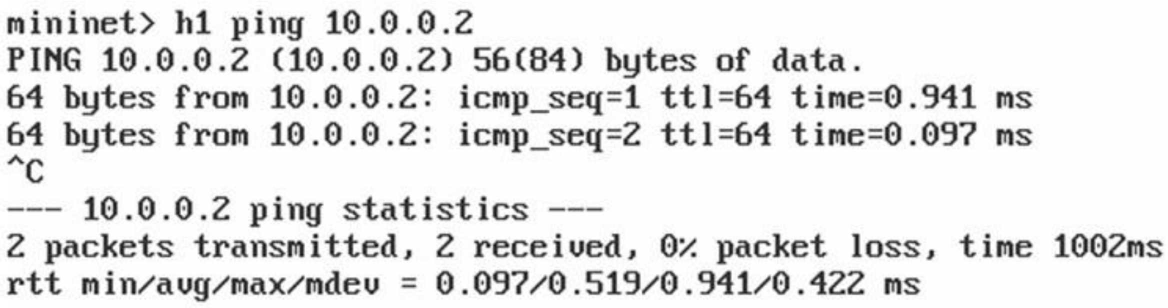
mininet> h1 ping -c1 h2
(Send 1 ping packet to host h2)

mininet> h1 ping 10.0.0.2
(From h1 ping to ip address “10.0.0.2”)

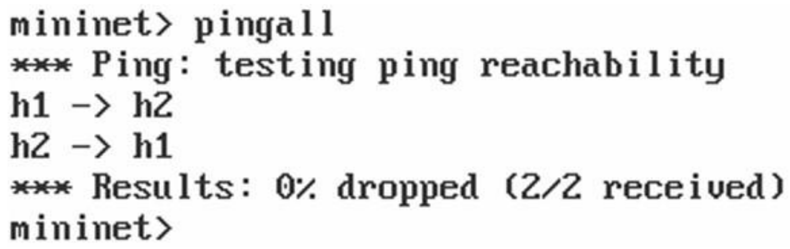
mininet> pingall
(All hosts ping to one another)

mininet> h1 python -m SimpleHTTPServer 80 &
(Launch python based “SimpleHTTPServer” listening on port 80 on host h1)

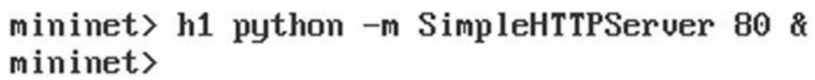
mininet> h2 wget -0 – h1
(verify that web server is working)



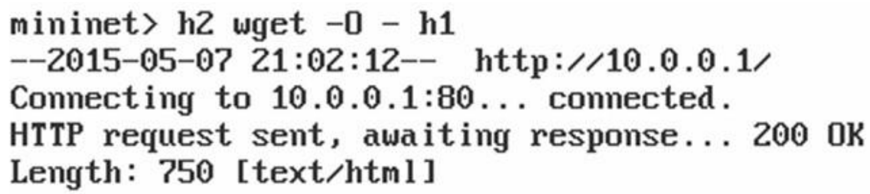
```
mininet> h1 ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.941 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.097 ms
^C
--- 10.0.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.097/0.519/0.941/0.422 ms
```



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

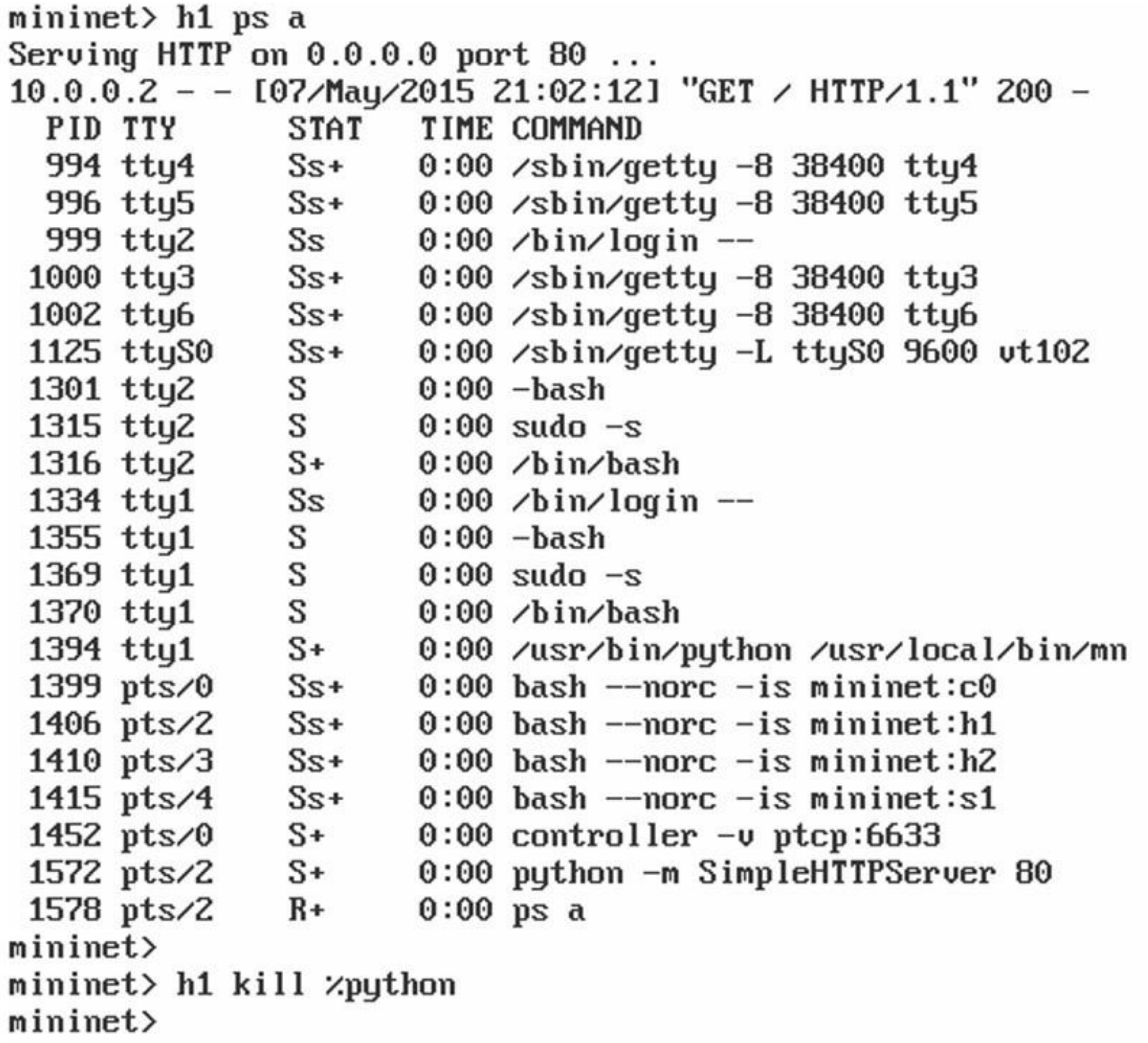


```
mininet> h1 python -m SimpleHTTPServer 80 &
mininet>
```



```
mininet> h2 wget -O - h1
--2015-05-07 21:02:12--  http://10.0.0.1/
Connecting to 10.0.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 750 [text/html]
```

Mininet Commands



```
mininet> h1 ps a
Serving HTTP on 0.0.0.0 port 80 ...
10.0.0.2 - - [07/May/2015 21:02:12] "GET / HTTP/1.1" 200 -
  PID TTY      STAT   TIME COMMAND
  994 tty4     Ss+    0:00 /sbin/getty -8 38400 tty4
  996 tty5     Ss+    0:00 /sbin/getty -8 38400 tty5
  999 tty2     Ss     0:00 /bin/login --
 1000 tty3     Ss+    0:00 /sbin/getty -8 38400 tty3
 1002 tty6     Ss+    0:00 /sbin/getty -8 38400 tty6
 1125 ttyS0    Ss+    0:00 /sbin/getty -L ttyS0 9600 vt102
 1301 tty2     S      0:00 -bash
 1315 tty2     S      0:00 sudo -s
 1316 tty2     S+     0:00 /bin/bash
 1334 tty1     Ss     0:00 /bin/login --
 1355 tty1     S      0:00 -bash
 1369 tty1     S      0:00 sudo -s
 1370 tty1     S      0:00 /bin/bash
 1394 tty1     S+     0:00 /usr/bin/python /usr/local/bin/mn
 1399 pts/0    Ss+    0:00 bash --norc -is mininet:c0
 1406 pts/2    Ss+    0:00 bash --norc -is mininet:h1
 1410 pts/3    Ss+    0:00 bash --norc -is mininet:h2
 1415 pts/4    Ss+    0:00 bash --norc -is mininet:s1
 1452 pts/0    S+     0:00 controller -v ptcp:6633
 1572 pts/2    S+     0:00 python -m SimpleHTTPServer 80
 1578 pts/2    R+     0:00 ps a
mininet>
mininet> h1 kill %python
mininet>
```

mininet> h1 ps a

(It will show all the process running on host h1. Sometimes we need to kill the processes. There are 2 ways to kill the process. One by giving the name of process & other by specifying pid)

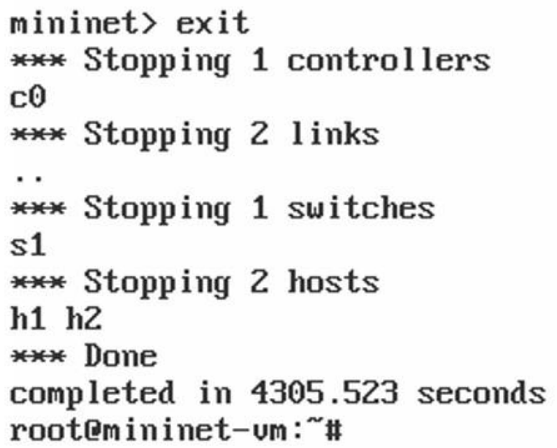
mininet> h1 kill %python

mininet> h1 kill -9 1572

(Kill the process by name or pid)

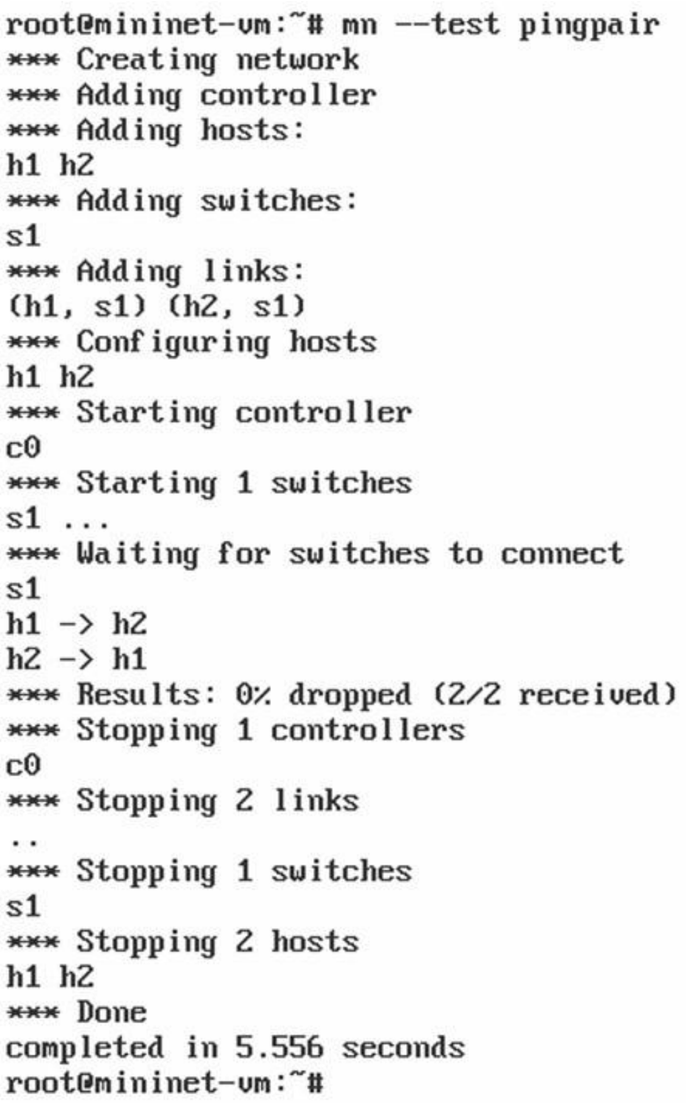
mininet> exit

(Exit from mininet prompt)



```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 4305.523 seconds
root@mininet-vm:~#
```

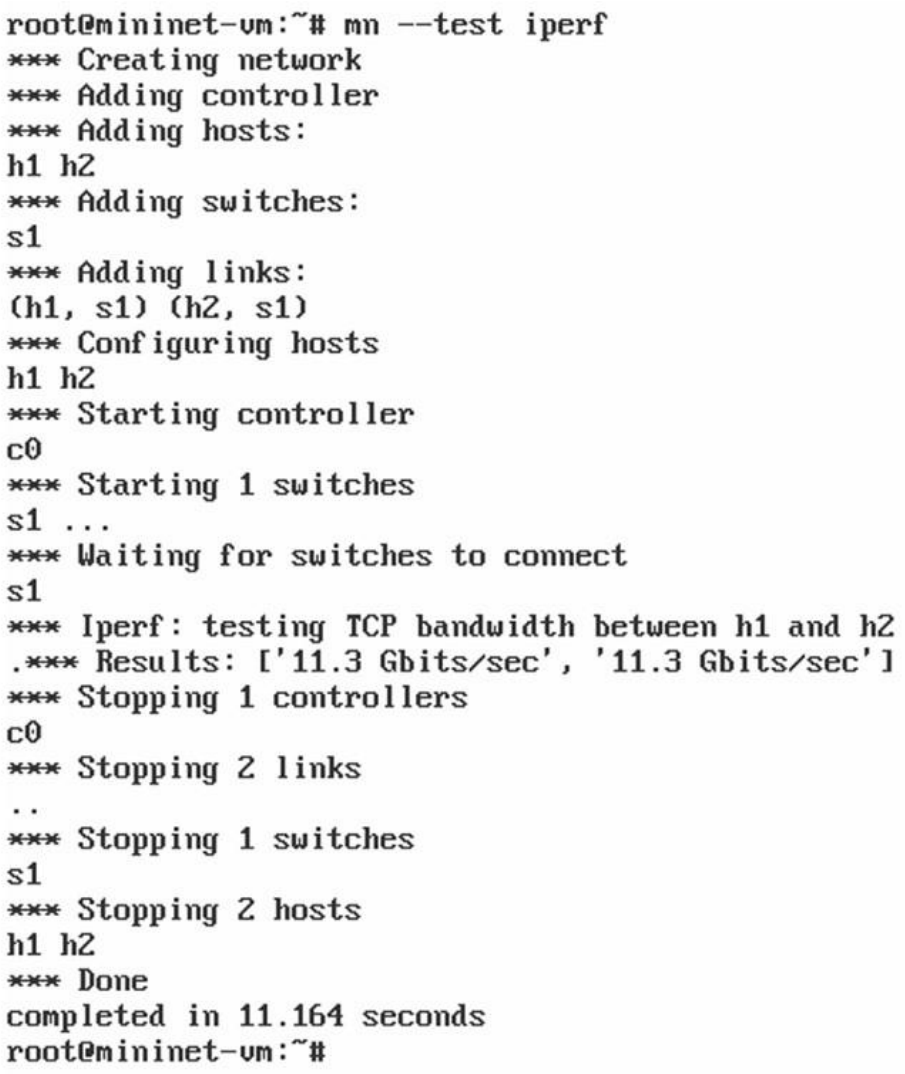
Mininet Commands



```
root@mininet-vm:~# mn --test pingpair
*** 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 ...
*** Waiting for switches to connect
s1
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 5.556 seconds
root@mininet-vm:~#
```

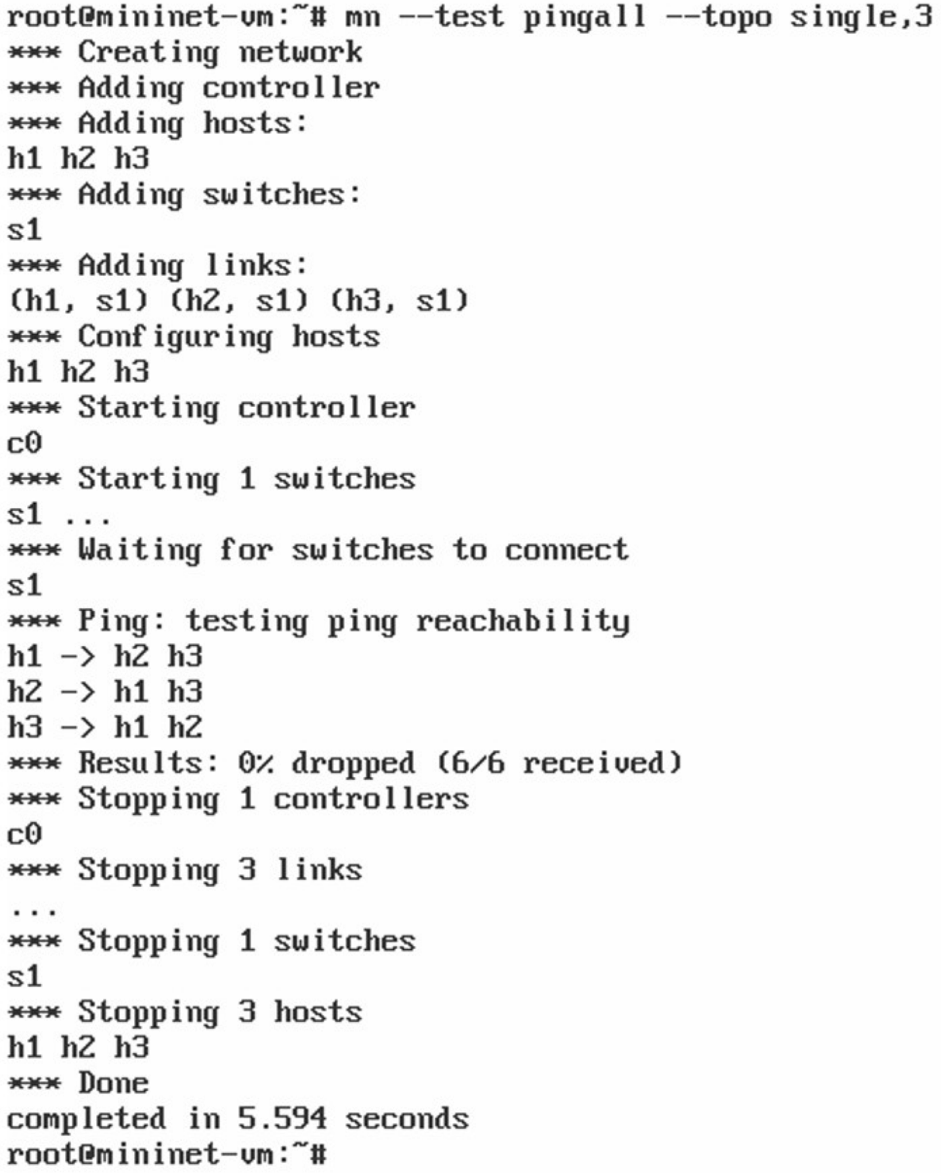
mn --test pingpair
(It will create network topology, perform ping test & then exit. Useful for testing purposes)

mn --test iperf
(Useful for performing bandwidth tests. Here it is showing bandwidth of “11.3 Gbits/sec”)



```
root@mininet-vm:~# mn --test iperf
*** 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 ...
*** Waiting for switches to connect
s1
*** Iperf: testing TCP bandwidth between h1 and h2
.*** Results: ['11.3 Gbits/sec', '11.3 Gbits/sec']
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 11.164 seconds
root@mininet-vm:~#
```

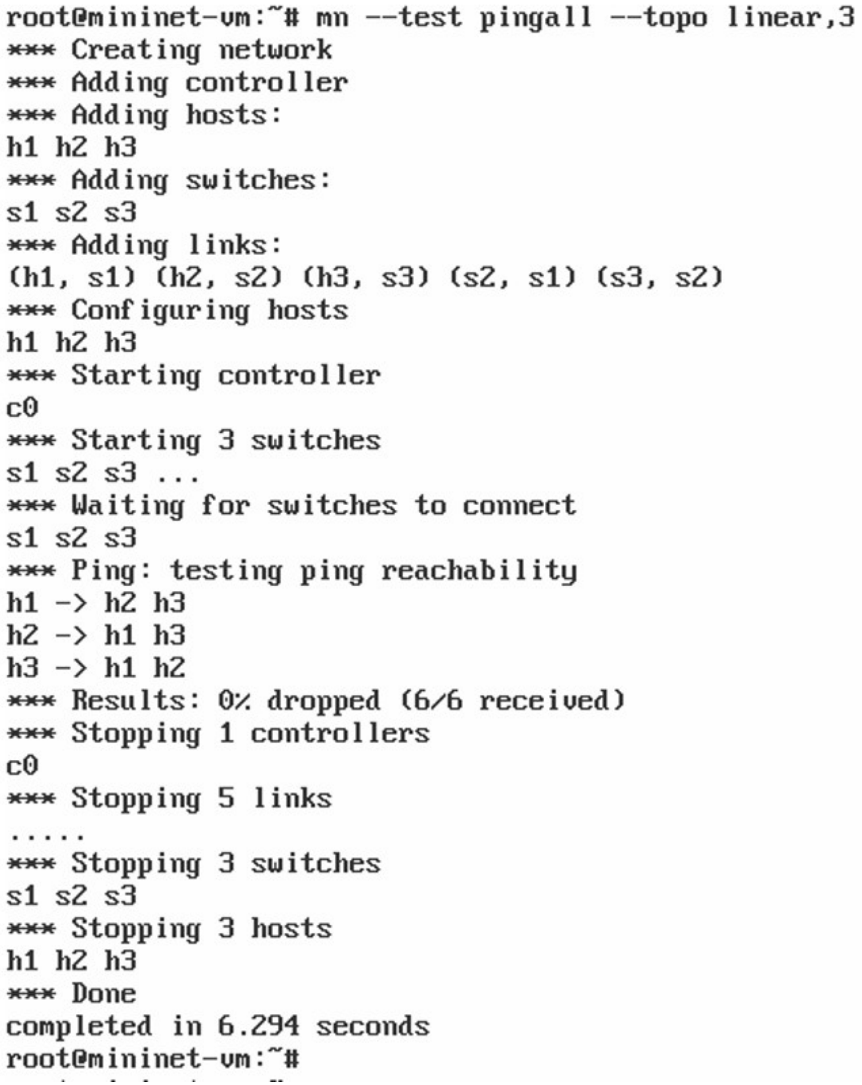
Mininet Commands



```
root@mininet-vm:~# mn --test pingall --topo single,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Waiting for switches to connect
s1
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (6/6 received)
*** Stopping 1 controllers
c0
*** Stopping 3 links
...
*** Stopping 1 switches
s1
*** Stopping 3 hosts
h1 h2 h3
*** Done
completed in 5.594 seconds
root@mininet-vm:~#
```

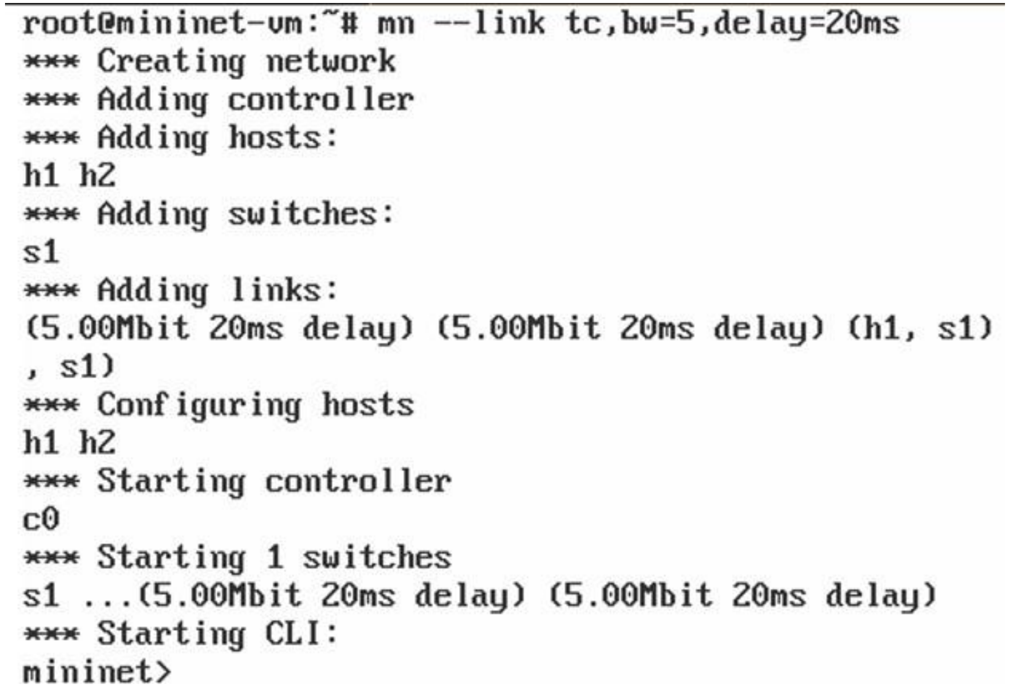
# mn --test pingall --topo single,3 (Create topology of 1 switch, 3 hosts, 1 reference controller. Do pingall test & exit)

# mn --test pingall --topo linear,3 (Create topology of 3 switch, 3 hosts, 1 reference controller. Do pingall test & exit)

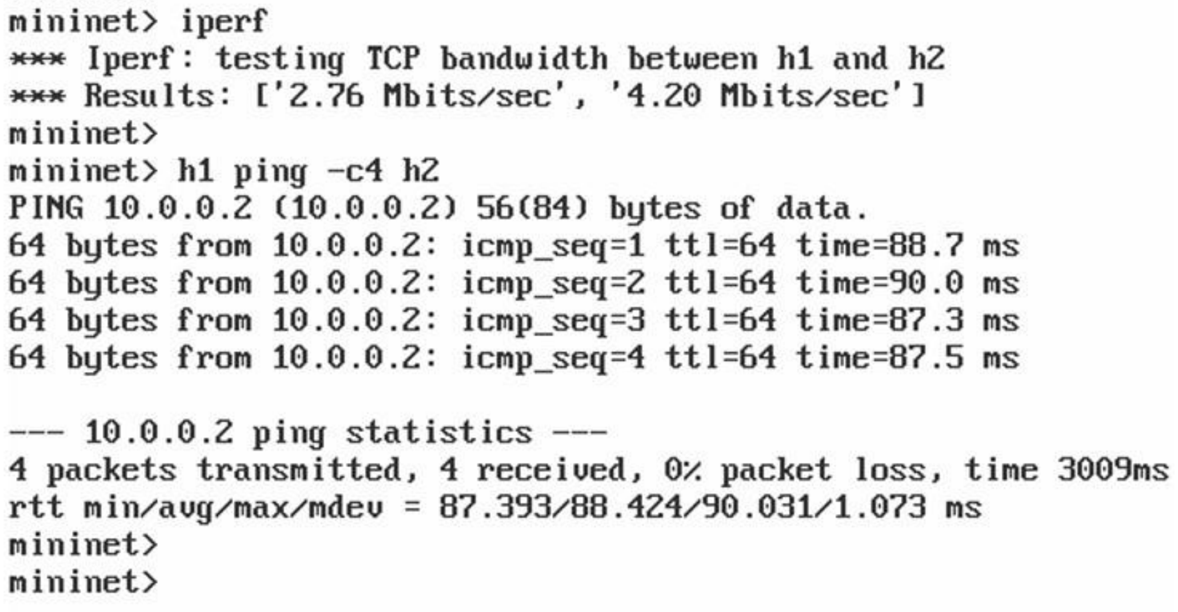


```
root@mininet-vm:~# mn --test pingall --topo linear,3
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (s2, s1) (s3, s2)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Waiting for switches to connect
s1 s2 s3
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 h3
h3 -> h1 h2
*** Results: 0% dropped (6/6 received)
*** Stopping 1 controllers
c0
*** Stopping 5 links
.....
*** Stopping 3 switches
s1 s2 s3
*** Stopping 3 hosts
h1 h2 h3
*** Done
completed in 6.294 seconds
root@mininet-vm:~#
```

Mininet Commands



```
root@mininet-vm:~# mn --link tc,bw=5,delay=20ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(5.00Mbit 20ms delay) (5.00Mbit 20ms delay) (h1, s1)
, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...(5.00Mbit 20ms delay) (5.00Mbit 20ms delay)
*** Starting CLI:
mininet>
```



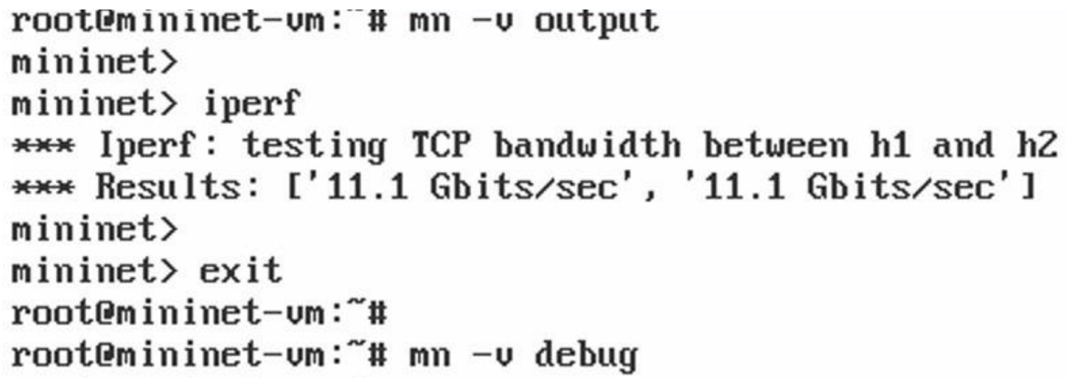
```
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['2.76 Mbits/sec', '4.20 Mbits/sec']
mininet>
mininet> h1 ping -c4 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=88.7 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=90.0 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=87.3 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=87.5 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 87.393/88.424/90.031/1.073 ms
mininet>
mininet>
```

mn --link tc,bw=5,delay=20ms

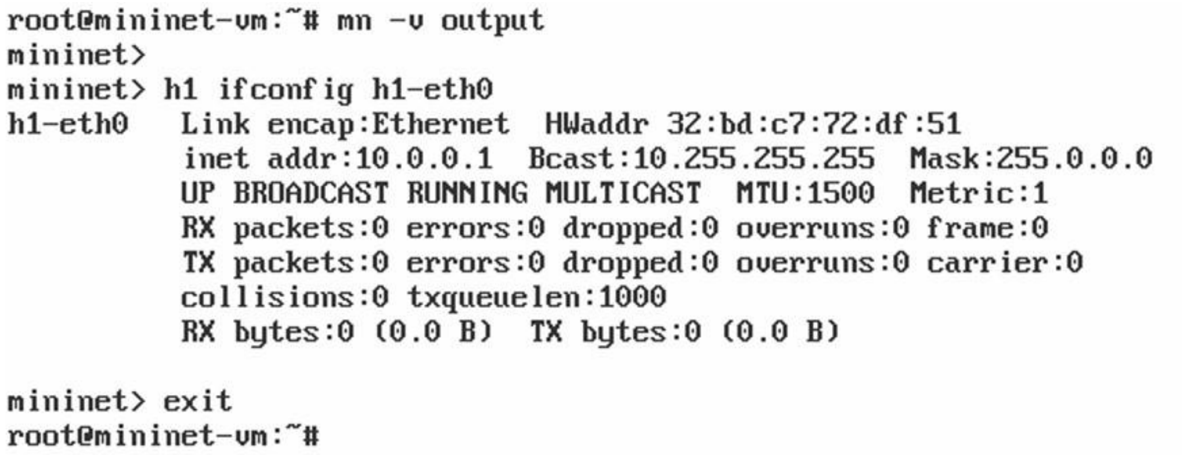
(Use --link option for traffic control (tc) & for specifying bandwidth (5MB) & delay of 20ms. Now on running iperf it is showing the bandwidth less than 5MB & delay between h1 & h2 is more than 80ms)

Why ? because rtt is = time from h1 to s1 (20ms) + time from s1 to h2 (20ms) + time from h2 to s1 (20ms) + time from s1 to h1 (20ms)

Mininet Commands



```
root@mininet-vm:~# mn -v output
mininet>
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h2
*** Results: ['11.1 Gbits/sec', '11.1 Gbits/sec']
mininet>
mininet> exit
root@mininet-vm:~#
root@mininet-vm:~# mn -v debug
```

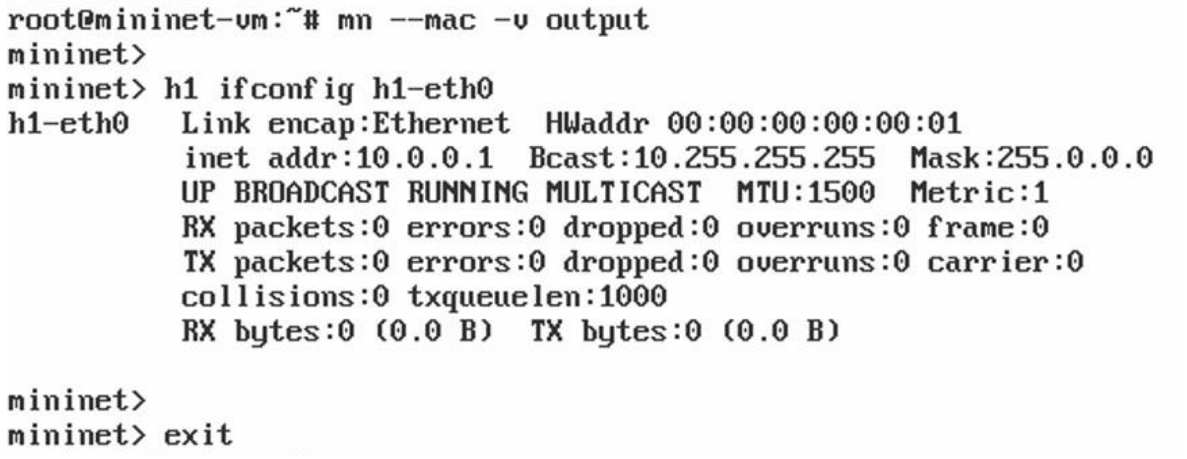


```
root@mininet-vm:~# mn -v output
mininet>
mininet> h1 ifconfig h1-eth0
h1-eth0   Link encap:Ethernet  HWaddr 32:bd:c7:72:df:51
          inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet> exit
root@mininet-vm:~#
```

# mn -v output
(It will takes you to mininet prompt without giving any messages)

# mn -v debug
(Will give you lot of information during topology creation & exit. Default verbosity is info)

what is the difference between running “mn” & “mn –mac”. Running with “--mac“ will generate easy to remember mac address corresponding to your ip address otherwise will generate difficult to remember random mac addresses.

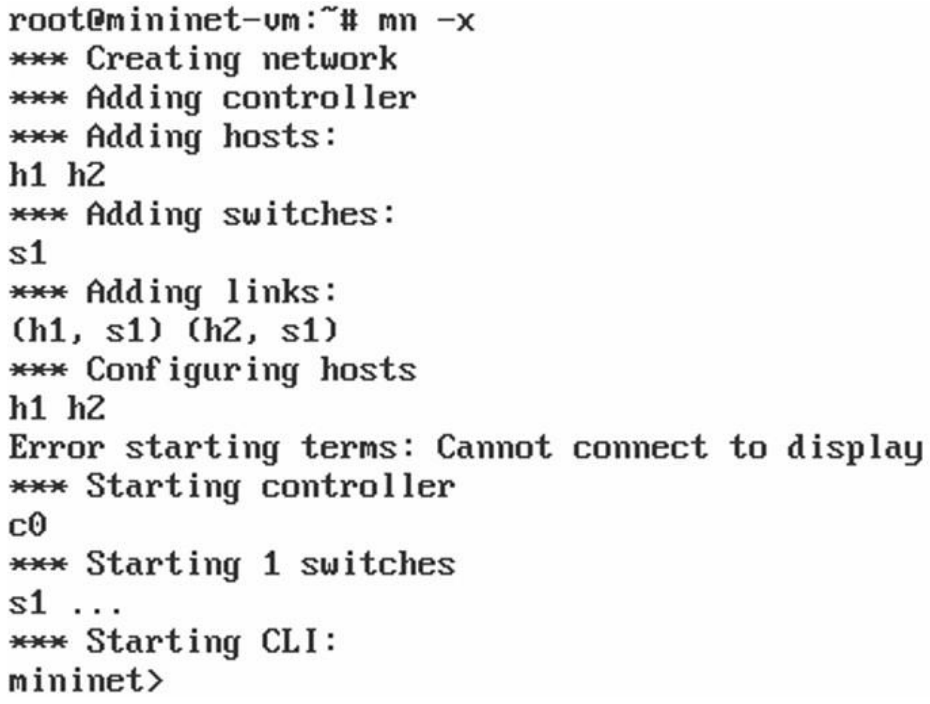


```
root@mininet-vm:~# mn --mac -v output
mininet>
mininet> h1 ifconfig h1-eth0
h1-eth0   Link encap:Ethernet  HWaddr 00:00:00:00:00:01
          inet addr:10.0.0.1  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

mininet>
mininet> exit
```

Mininet Commands

# mn -x

(Will open x terminal. You need to run this command on GUI terminal otherwise it will give error "can not connect to display" as it is giving in our case)



```
root@mininet-vm:~# mn -x
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
Error starting terms: Cannot connect to display
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```