Lab 0 — Mininet

Team: Alwyn Johnson

Repository: https://github.com/alwyn090/5718-Lab-Assignments

Video: https://youtu.be/2W3jIpbpSyw

Course/Lab: EEL5718 Lab 0

Date: 09/09/2025

1) Definitions & Pre-Lab Concepts

1.a Repository Link:

https://github.com/alwyn090/5718-Lab-Assignments

b) Linux shell, root shell

Linux shell is a command-line interpreter that allows users to interact with the operating system by typing commands (Bash).

Root shell is A shell session with superuser (administrator) privileges, giving full control over the system.

c) Wireshark

Wireshark is a network protocol analyzer that captures and inspects packets in real time, useful for troubleshooting and studying network behavior.

d) sudo

Is short for “superuser do” allows a permitted user to run commands with elevated privileges, usually required for administrative tasks.

e) virtual machine (VM)

A VM is a software-based emulation of a physical computer that runs an operating system and applications independently of the host machine.

f) ifconfig

ifconfig is a command-line tool used to configure and display network interface settings, such as IP addresses and hardware (MAC) addresses.

g) arp command, arp protocol

arp command displays and modifies the system’s ARP (Address Resolution Protocol) table, mapping IP addresses to MAC addresses.

ARP protocol is network protocol used to translate an IP address into a corresponding MAC address within a local network.

h) network interface

A network interface is the point of interconnection between a computer and a network, either physical (e.g., Ethernet card) or virtual (e.g., eth0, wlan0).

i) xterm terminal emulator
xterm is a terminal emulator for the X Window System that provides a command-line interface window to run shell sessions.

j) Command Line Interface (CLI)
A CLI is a text-based interface where users interact with software or the operating system by typing commands, as opposed to using a graphical interface.

k) Describe the difference between a host, a switch and a controller in Mininet.
A Host represents an end device (like a computer) that can send and receive traffic.
A Switch Simulates a virtual switch that forwards packets between hosts using MAC addresses. Controller is a centralized program in Software-Defined Networking (SDN) that manages and configures switches' behavior.

---

2) Part 1 — Everyday Mininet Usage

2.a Interact with Hosts and Switches

i) Sketch the default topology:

The default 'mn' builds h1—s1—h2 with a reference controller if available. Label h1, h2, s1, controller; draw links h1–s1 and h2–s1. (See Fig. 1.)

ii) ifconfig outputs (screenshots):

h1 ifconfig

h2 ifconfig

s1 ifconfig

Include screenshots showing IP/MAC per node. (Figs. 2–4)

iii) arp and route on s1 and h1 with discussion:

h1 arp

h1 route

s1 arp

s1 route

Observation: Each node maintains its own neighbor (ARP) and routing tables. Initially h1 does not list h2 in ARP until traffic prompts resolution (e.g., ping). s1's control-plane state differs from host tables, demonstrating isolated network state across nodes. (Figs. 5)

2.b Test Connectivity Between Hosts

```
h1 ping -c 1 h2
```

Provide ping screenshots and summarize loss/RTT. (Fig. 6)

2.c Run a Simple Web Server and Client

Verify Python/Mininet versions:

```
mn --version

python3 --version

mininet> h1 python3 -m http.server 80 &

mininet> h2 wget -O - h1

...

mininet> h1 kill %python
```

Screenshots should show version strings, HTTP 200 response, and bytes received. Record your Mininet version and Python version explicitly. (Figs. 7–9)

---

3) Part 2 — Advanced Startup Options

3.a Changing Topology Size and Type

i) One switch, six hosts; ping h1→h3:

```
sudo mn --topo single,6

h1 ping -c 3 h3
```

Capture command and output (Fig. 10).

ii) Linear, six switches (one host per switch); ping h1→h3:

```
sudo mn --topo linear,6

h1 ping -c 3 h3
```

Capture command and output (Fig. 11).

iii) Which has higher latency, and why?

The **linear topology** will have a longer observed latency than the single-switch topology. **Reason:** In the linear case, packets from h1 to h3 must traverse multiple switches (s1 → s2 → s3), while in the single-switch case, all hosts are directly connected to the same switch. Each additional switch introduces processing and forwarding delay. ping statistics in Figs. 10–11..

3.b Link Variations — RTT with 20 ms per link

Launch with traffic control (tc) link delay and measure using ping:

```
sudo mn --topo single,6 --link tc,delay=20ms

h1 ping -c 3 h3

sudo mn -c

sudo mn --topo linear,6 --link tc,delay=20ms

h1 ping -c 3 h3
```

Back-of-envelope: RTT ≈ 2 × (links in one-way path) × 20 ms. Single-switch (h1–s1–h3): 2 links ⇒ RTT ≈ 80 ms. Linear (h1 on s1, h3 on s3): 4 links ⇒ RTT ≈ 160 ms. Compare to your observed ping RTTs. (Figs. 12–13)

3.c Custom Topologies

i) Analyze topo-2sw-2host.py and show pingall output:

```
sudo mn --custom ~/Downloads/mininet/custom/topo-2sw-2host.py --topo mytopo --test pingall
```

Discussion: Two hosts (h1, h2) each attach to separate switches (s1, s2) with an inter-switch link s1–s2. pingall should succeed across the two-switch path (h1↔h2). Include build logs and ping results. (Fig. 14)

References

Mininet Walkthrough (Parts 1–4) — http://mininet.org/walkthrough/

Figures

Figure 1. Default Mininet topology sketch (h1—s1—h2, reference controller).

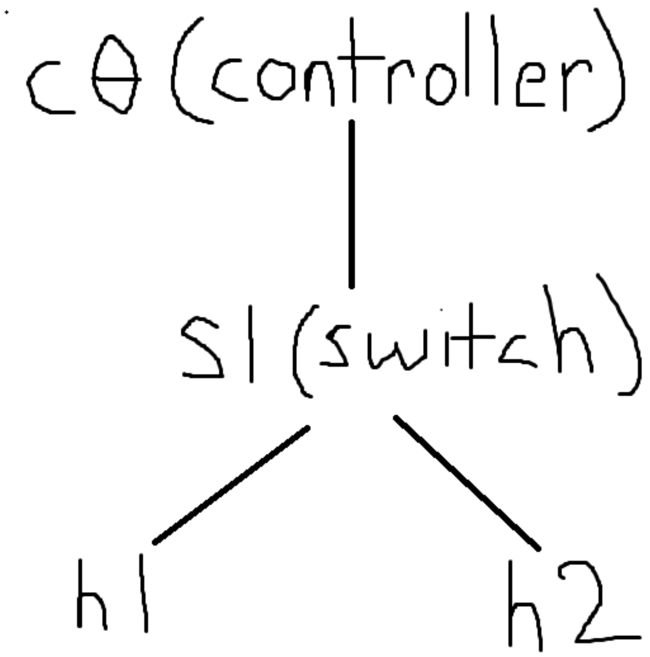
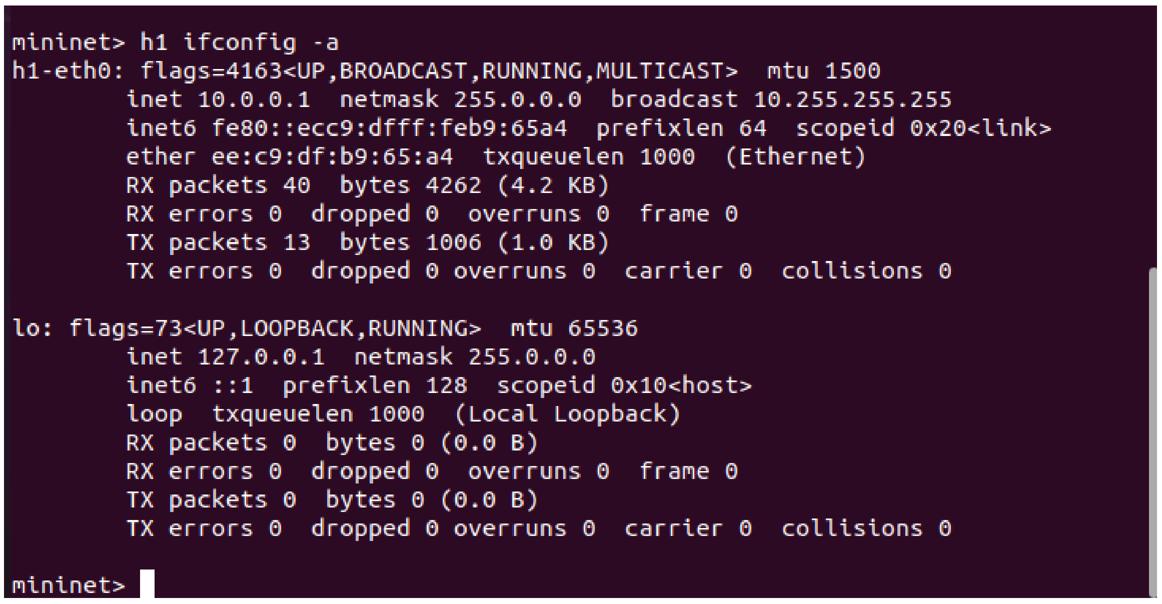
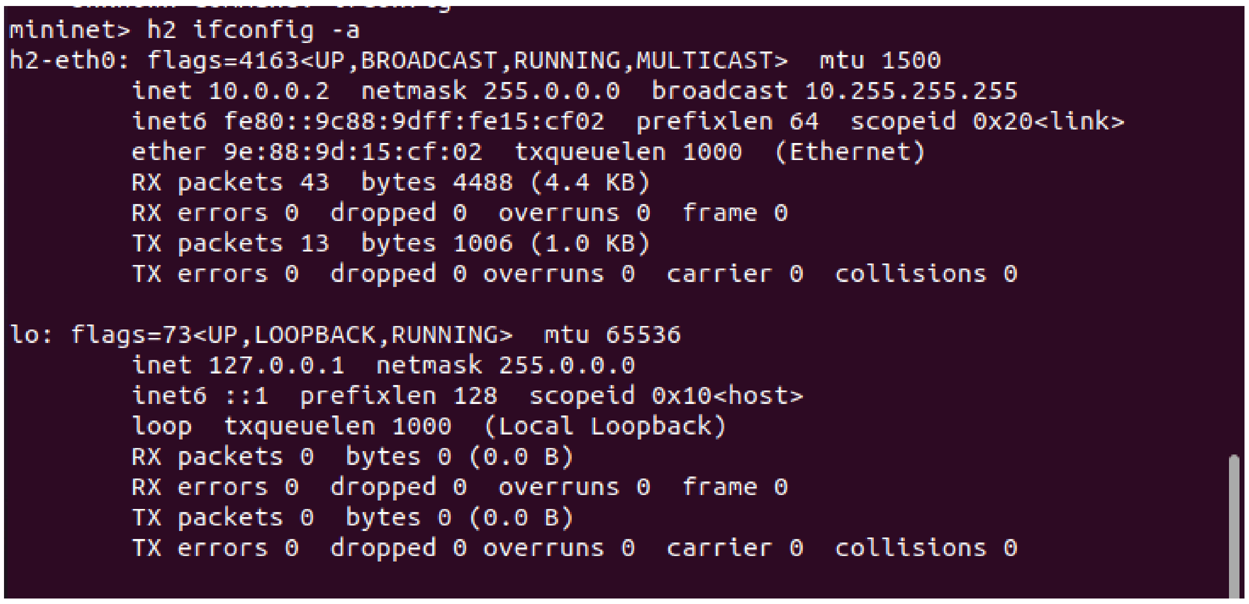


Figure 2. h1 ifconfig output (interfaces, IP/MAC).



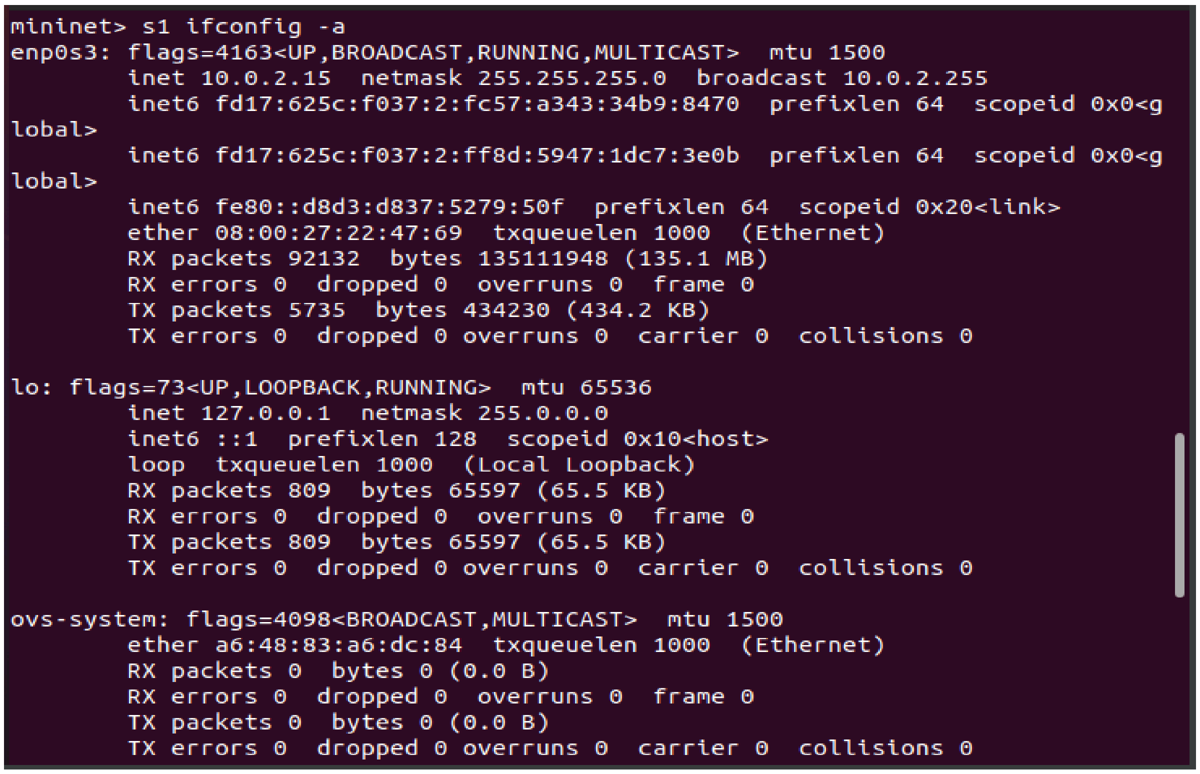
```
mininet> h1 ifconfig -a
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::ecc9:dfff:feb9:65a4  prefixlen 64  scopeid 0x20<link>
        ether ee:c9:df:b9:65:a4  txqueuelen 1000  (Ethernet)
        RX packets 40  bytes 4262 (4.2 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> 
```

Figure 3. h2 ifconfig output.

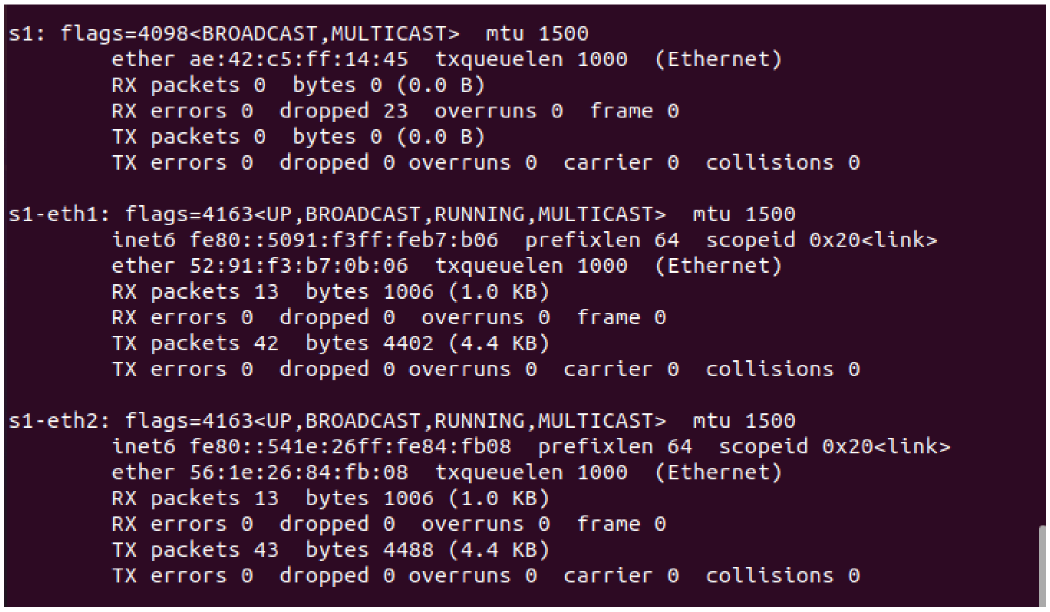


```
mininet> h2 ifconfig -a
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.2  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::9c88:9dff:fe15:cf02  prefixlen 64  scopeid 0x20<link>
        ether 9e:88:9d:15:cf:02  txqueuelen 1000  (Ethernet)
        RX packets 43  bytes 4488 (4.4 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
```

Figure 4. s1 ifconfig output (ports/links).

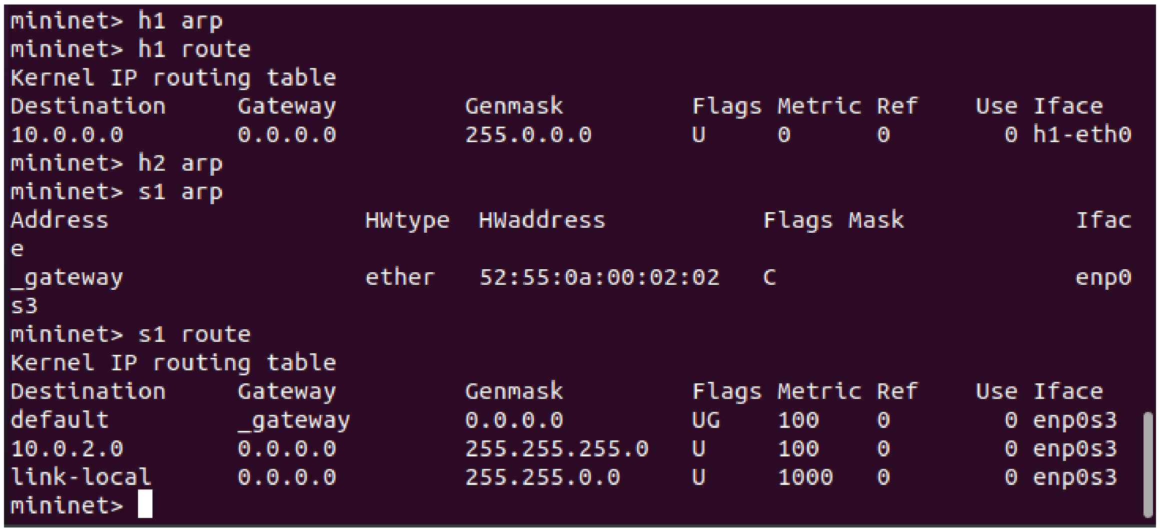


```
mininet> s1 ifconfig -a
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fd17:625c:f037:2:fc57:a343:34b9:8470  prefixlen 64  scopeid 0x0<g
lobal>
        inet6 fd17:625c:f037:2:ff8d:5947:1dc7:3e0b  prefixlen 64  scopeid 0x0<g
lobal>
        inet6 fe80::d8d3:d837:5279:50f  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:22:47:69  txqueuelen 1000  (Ethernet)
        RX packets 92132  bytes 135111948 (135.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5735  bytes 434230 (434.2 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 809  bytes 65597 (65.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 809  bytes 65597 (65.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ovs-system: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether a6:48:83:a6:dc:84  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
```



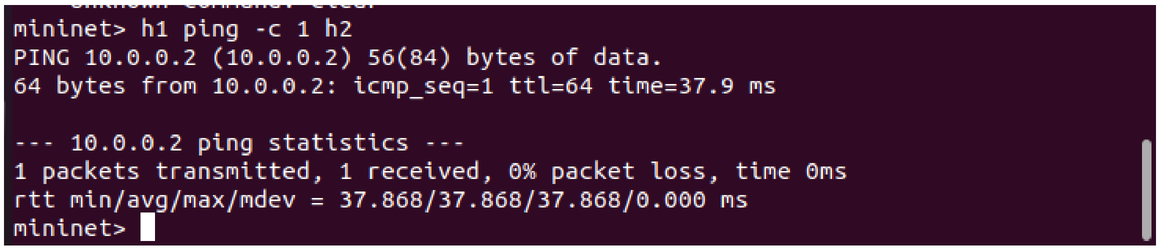
```
s1: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether ae:42:c5:ff:14:45  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 23  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::5091:f3ff:feb7:b06  prefixlen 64  scopeid 0x20<link>
        ether 52:91:f3:b7:0b:06  txqueuelen 1000  (Ethernet)
        RX packets 13  bytes 1006 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 42  bytes 4402 (4.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::541e:26ff:fe84:fb08  prefixlen 64  scopeid 0x20<link>
        ether 56:1e:26:84:fb:08  txqueuelen 1000  (Ethernet)
        RX packets 13  bytes 1006 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 43  bytes 4488 (4.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 5. arp route

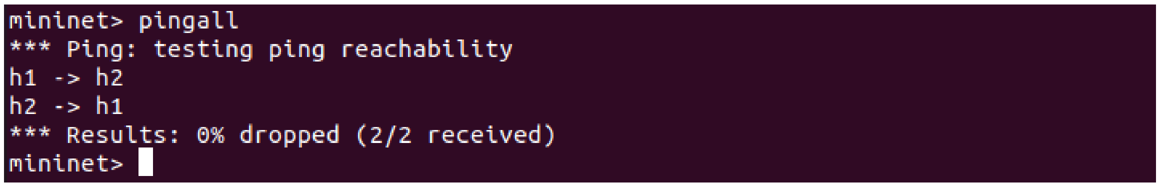


```
mininet> h1 arp
mininet> h1 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 h1-eth0
mininet> h2 arp
mininet> s1 arp
Address                  HWtype  HWaddress           Flags Mask            Ifac
e
_gateway                 ether   52:55:0a:00:02:02   C                     enp0
s3
mininet> s1 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
link-local      0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
mininet> 
```

Figure 6. h1 ping -c 1 h2 — connectivity and RTT summary.

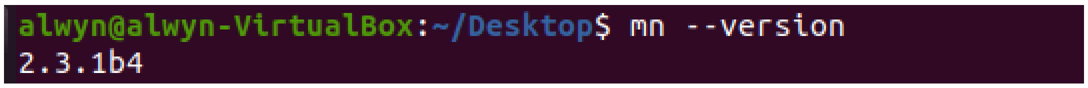


```
mininet> h1 ping -c 1 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=37.9 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 37.868/37.868/37.868/0.000 ms
mininet> 
```



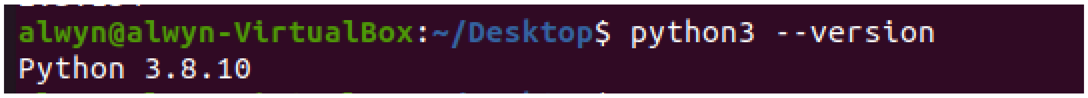
```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
mininet> 
```

Figure 7. mn --version output (Mininet version).

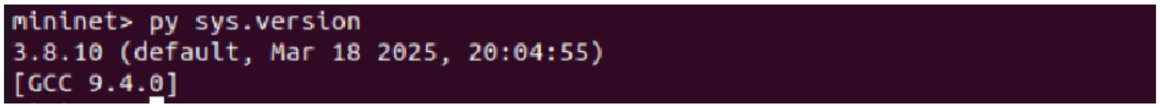


```
alwyn@alwyn-VirtualBox:~/Desktop$ mn --version
2.3.1b4
```

Figure 8. python3 --version output.

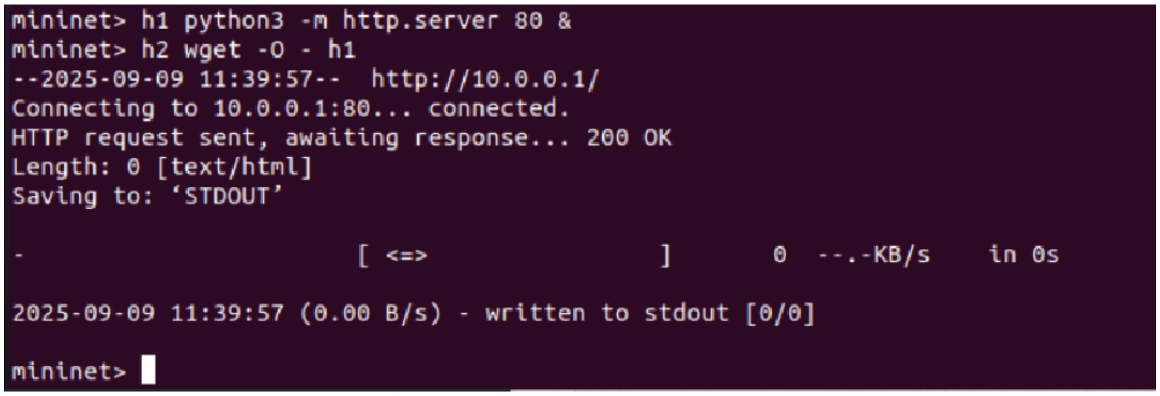


```
alwyn@alwyn-VirtualBox:~/Desktop$ python3 --version
Python 3.8.10
```

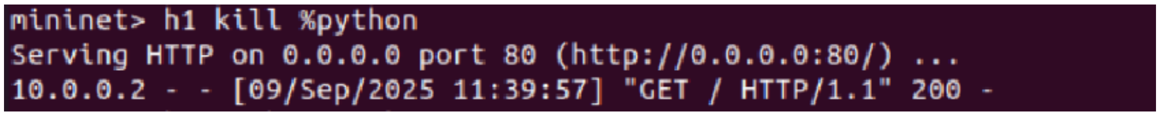


```
mininet> py sys.version
3.8.10 (default, Mar 18 2025, 20:04:55)
[GCC 9.4.0]
```

Figure 9. Web demo — server (h1 http.server) and client fetch (h2 wget) with HTTP 200.

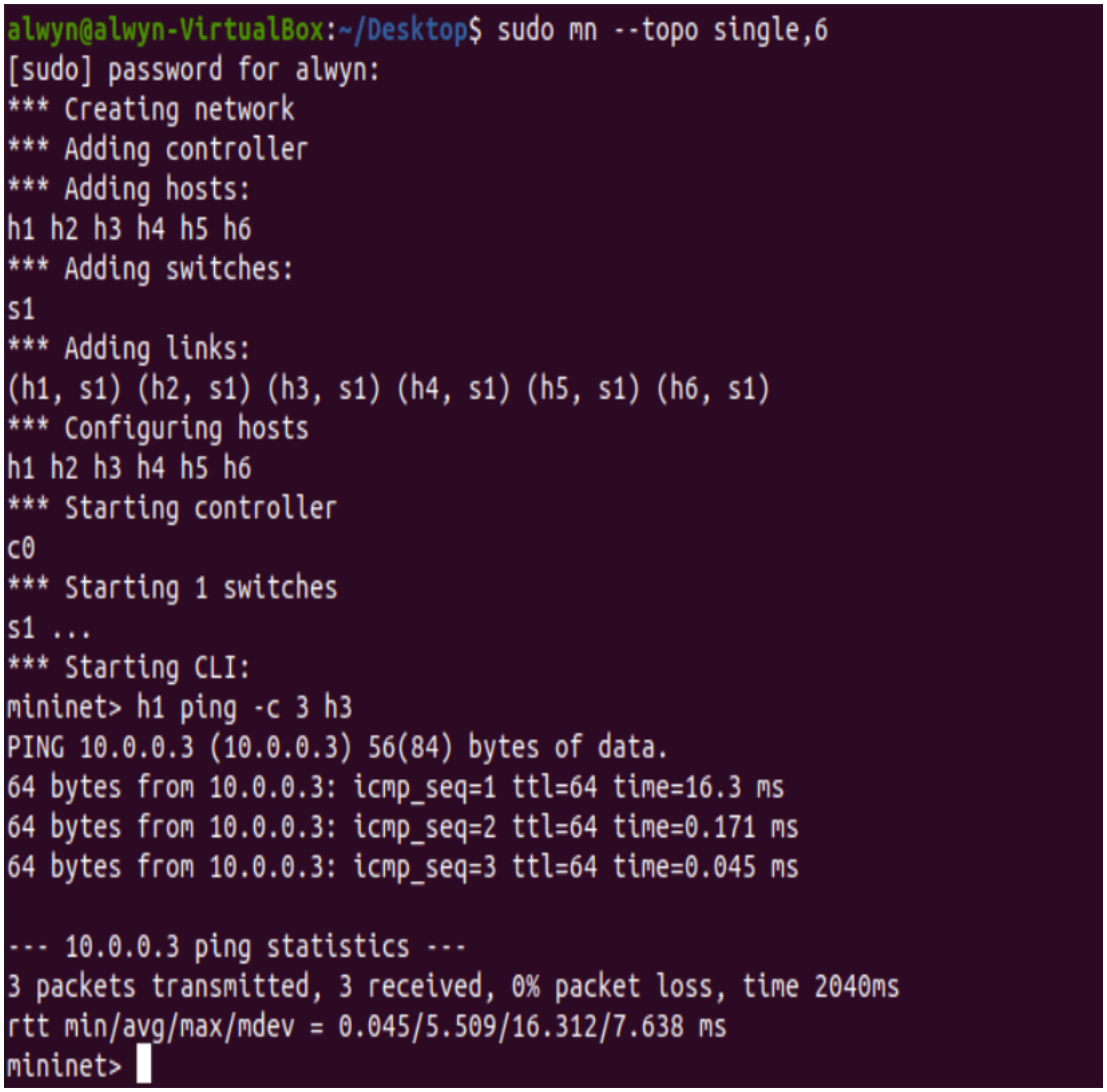


```
mininet> h1 python3 -m http.server 80 &
mininet> h2 wget -O - h1
--2025-09-09 11:39:57--  http://10.0.0.1/
Connecting to 10.0.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/html]
Saving to: ‘STDOUT’

-                       [ <=>                ]       0  --.-KB/s    in 0s

2025-09-09 11:39:57 (0.00 B/s) - written to stdout [0/0]

mininet> 
```



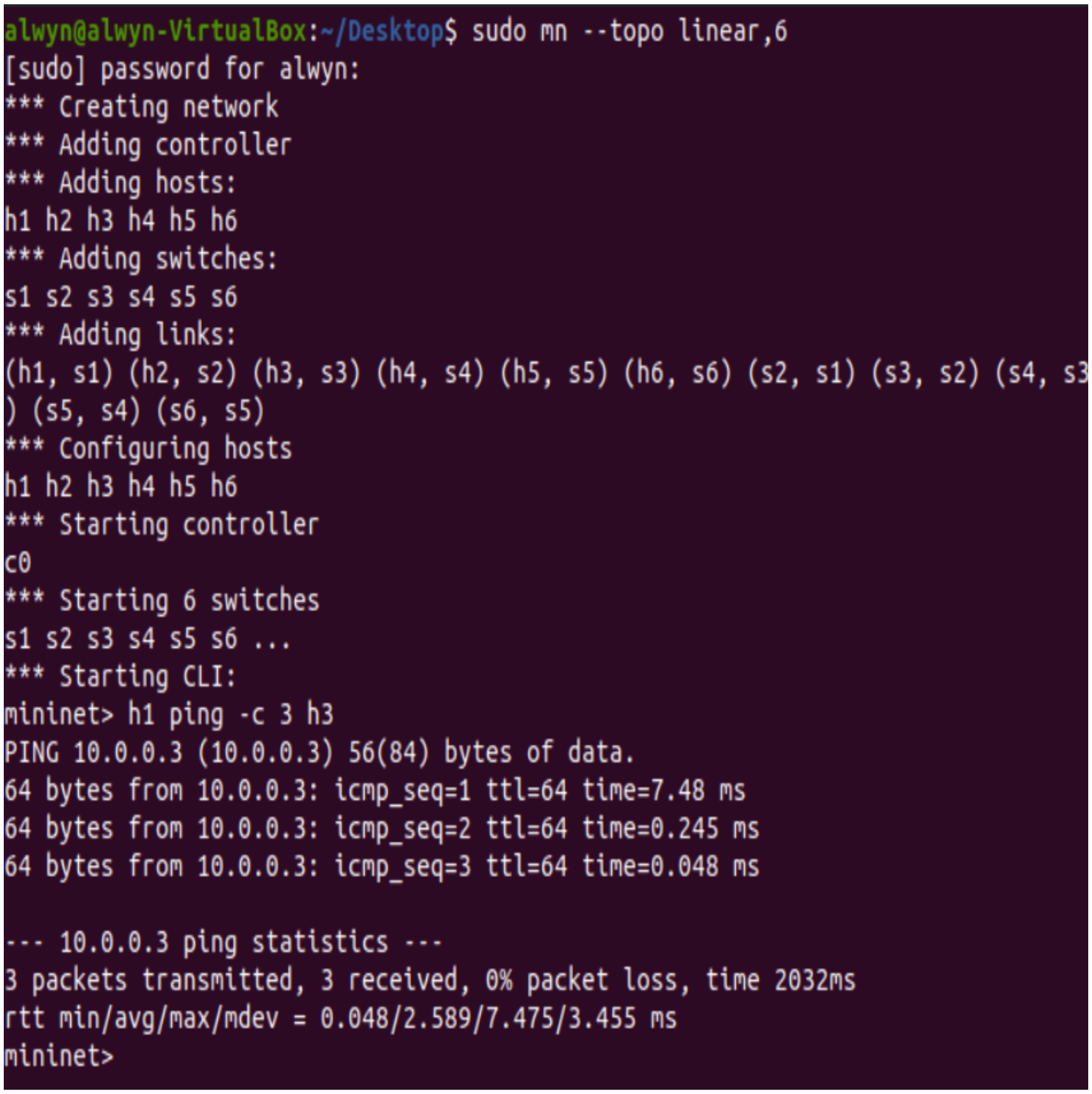
```
mininet> h1 kill %python
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.0.0.2 - - [09/Sep/2025 11:39:57] "GET / HTTP/1.1" 200 -
```

Figure 10. single,6 — h1 ping -c 3 h3 output.



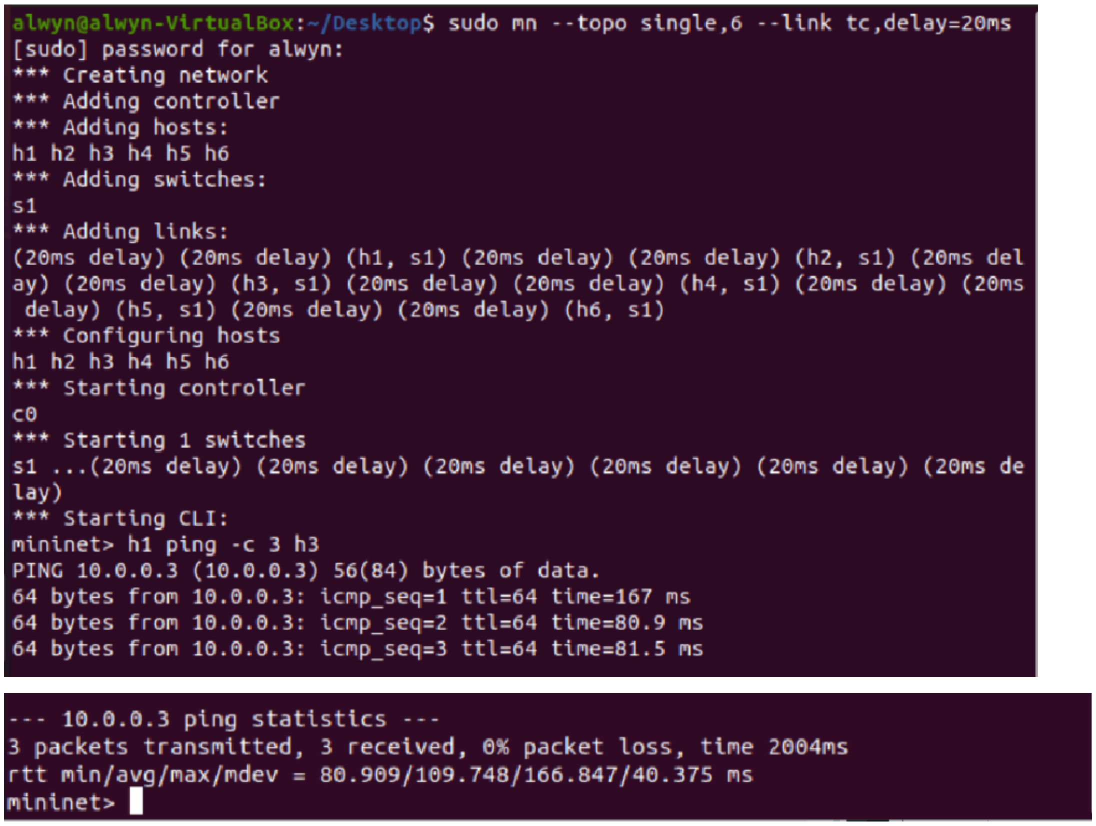
```
alwyn@alwyn-VirtualBox:~/Desktop$ sudo mn --topo single,6
[sudo] password for alwyn:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1) (h3, s1) (h4, s1) (h5, s1) (h6, s1)
*** Configuring hosts
h1 h2 h3 h4 h5 h6
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> h1 ping -c 3 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=16.3 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.171 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.045 ms

--- 10.0.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2040ms
rtt min/avg/max/mdev = 0.045/5.509/16.312/7.638 ms
mininet>
```

Figure 11. linear,6 — h1 ping -c 3 h3 output.



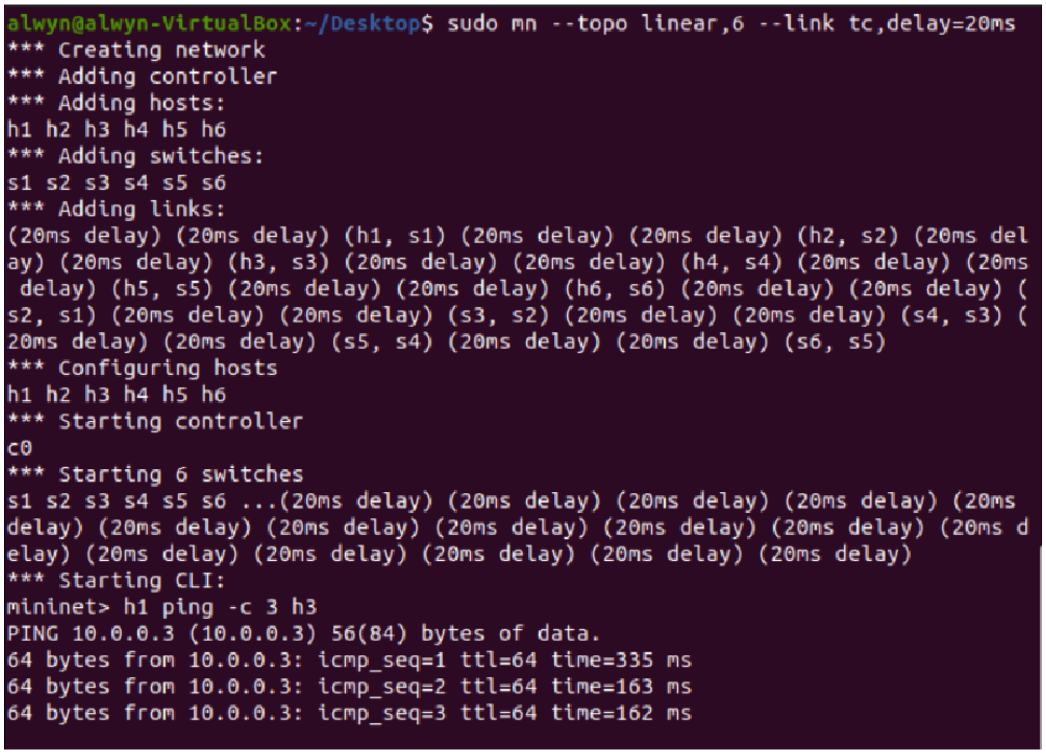
```
alwyn@alwyn-VirtualBox:~/Desktop$ sudo mn --topo linear,6
[sudo] password for alwyn:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6
*** Adding switches:
s1 s2 s3 s4 s5 s6
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (h5, s5) (h6, s6) (s2, s1) (s3, s2) (s4, s3
) (s5, s4) (s6, s5)
*** Configuring hosts
h1 h2 h3 h4 h5 h6
*** Starting controller
c0
*** Starting 6 switches
s1 s2 s3 s4 s5 s6 ...
*** Starting CLI:
mininet> h1 ping -c 3 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=7.48 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.245 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.048 ms

--- 10.0.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2032ms
rtt min/avg/max/mdev = 0.048/2.589/7.475/3.455 ms
mininet>
```

Figure 12. single,6 with --link tc,delay=20ms — ping RTT screenshot.

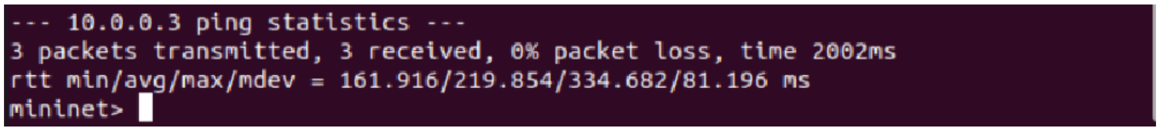


```
alwyn@alwyn-VirtualBox:~/Desktop$ sudo mn --topo single,6 --link tc,delay=20ms
[sudo] password for alwyn:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6
*** Adding switches:
s1
*** Adding links:
(20ms delay) (20ms delay) (h1, s1) (20ms delay) (20ms delay) (h2, s1) (20ms del
ay) (20ms delay) (h3, s1) (20ms delay) (20ms delay) (h4, s1) (20ms delay) (20ms
 delay) (h5, s1) (20ms delay) (20ms delay) (h6, s1)
*** Configuring hosts
h1 h2 h3 h4 h5 h6
*** Starting controller
c0
*** Starting 1 switches
s1 ...(20ms delay) (20ms delay) (20ms delay) (20ms delay) (20ms delay) (20ms de
lay)
*** Starting CLI:
mininet> h1 ping -c 3 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=167 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=80.9 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=81.5 ms

--- 10.0.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 80.909/109.748/166.847/40.375 ms
mininet>
```

Figure 13. linear,6 with --link tc,delay=20ms — ping RTT screenshot.

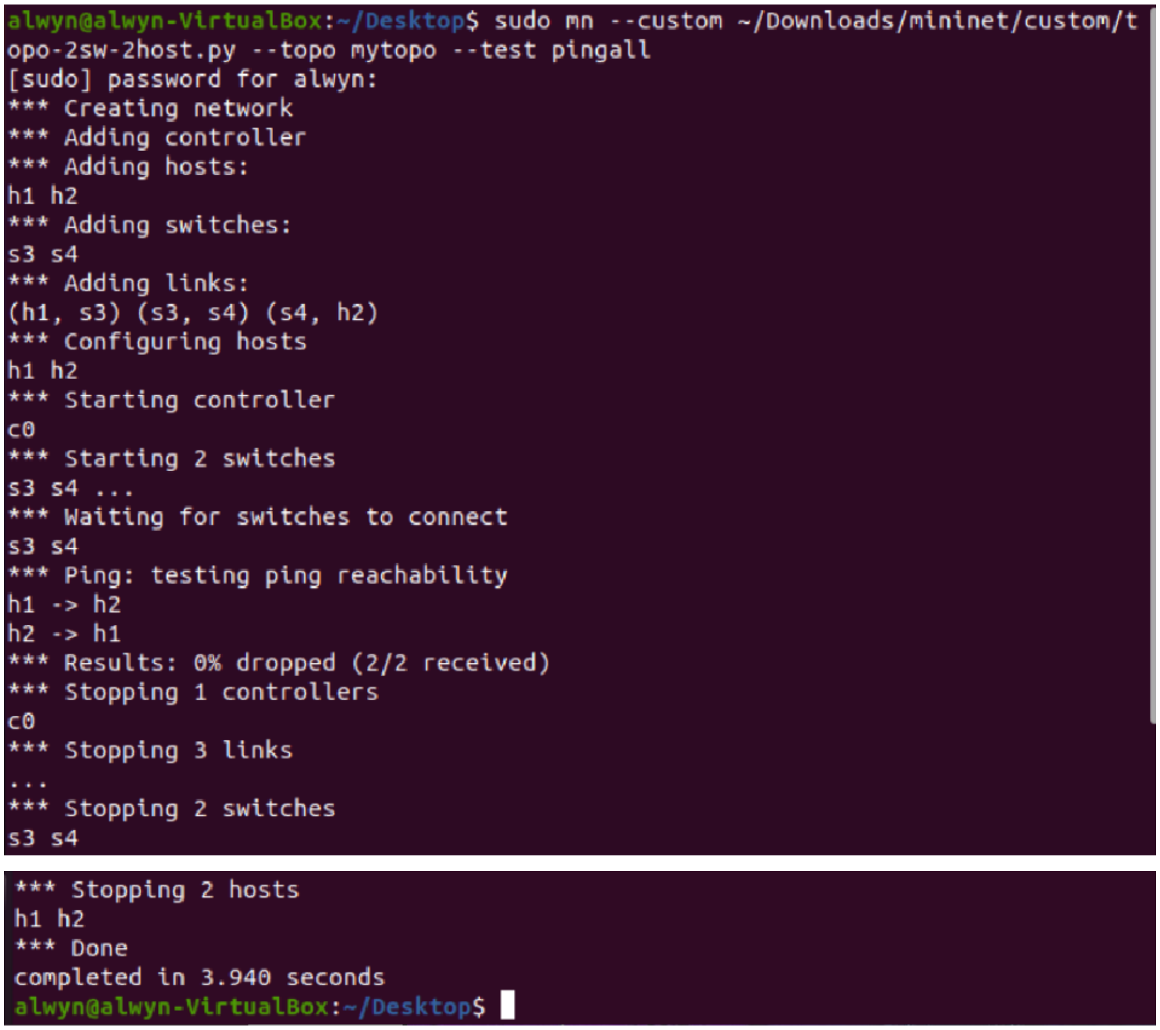


```
alwyn@alwyn-VirtualBox:~/Desktop$ sudo mn --topo linear,6 --link tc,delay=20ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6
*** Adding switches:
s1 s2 s3 s4 s5 s6
*** Adding links:
(20ms delay) (20ms delay) (h1, s1) (20ms delay) (20ms delay) (h2, s2) (20ms del
ay) (20ms delay) (h3, s3) (20ms delay) (20ms delay) (h4, s4) (20ms delay) (20ms
 delay) (h5, s5) (20ms delay) (20ms delay) (h6, s6) (20ms delay) (20ms delay) (
s2, s1) (20ms delay) (20ms delay) (s3, s2) (20ms delay) (20ms delay) (s4, s3) (
20ms delay) (20ms delay) (s5, s4) (20ms delay) (20ms delay) (s6, s5)
*** Configuring hosts
h1 h2 h3 h4 h5 h6
*** Starting controller
c0
*** Starting 6 switches
s1 s2 s3 s4 s5 s6 ...(20ms delay) (20ms delay) (20ms delay) (20ms delay) (20ms
delay) (20ms delay) (20ms delay) (20ms delay) (20ms delay) (20ms delay) (20ms d
elay) (20ms delay) (20ms delay) (20ms delay) (20ms delay) (20ms delay)
*** Starting CLI:
mininet> h1 ping -c 3 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=335 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=163 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=162 ms
```



```
--- 10.0.0.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 161.916/219.854/334.682/81.196 ms
mininet> 
```

Figure 14. Custom topo (topo-2sw-2host.py) — build and pingall results.



```
alwyn@alwyn-VirtualBox:~/Desktop$ sudo mn --custom ~/Downloads/mininet/custom/t
opo-2sw-2host.py --topo mytopo --test pingall
[sudo] password for alwyn:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s3 s4
*** Adding links:
(h1, s3) (s3, s4) (s4, h2)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s3 s4 ...
*** Waiting for switches to connect
s3 s4
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
*** Stopping 1 controllers
c0
*** Stopping 3 links
...
*** Stopping 2 switches
s3 s4
*** Stopping 2 hosts
h1 h2
*** Done
completed in 3.940 seconds
alwyn@alwyn-VirtualBox:~/Desktop$ 
```