Devin Dykstra Lab 0 Report

Team: Devin Dykstra (No other teammates)

1. Define/Explain the following terms:
    a. https://github.com/DevTechS/EEL5718_LabDemoRecordings A Repository folder is a directory that stores files for a project. Normally under a version control system like Git. In this case its made to store the demo recordings from the class labs
    b. The Linux shell is a command line interface that allows users to interact with the underlying Linux OS. The Root shell is the same thing, but with elevated administrator privileges.
    c. Wireshark is a free and open source tool used to analyze protocols and inspect data packets traveling over a network
    d. Sudo, short for "Supervised Do" allows a command to run with elevated administrator privileges.
    e. A Virtual Machine is a software system designed to emulate a hardware system. Instead of a physical machine, its virtual!
    f. Ifconfig is a command that displays useful network information on Linux.
    g. ARP (Address Resolution Protocol). Describes how IP addresses are translated to MAC addresses. An ARP command is used to control a system's ARP table.
    h. A network interface is the point where two systems interact on a network. Can be physical or virtual.
    i. The xterm terminal emulator is what allows command line interfaces to function, named off of the X Window System that is used in many unix operating systems.
    j. The Command line interface, as opposed to a Graphical User Interface, is where a user interacts with a system via text and commands rather than a mouse and buttons.
    k. .
        i. A Host is an end device like a computer or server
        ii. A switch forwards packets between host based on the flow rules
        iii. The controller actually establishes the flow and forwarding rules
2. Part 1: Everyday Mininet Usage
    a. .
        i. The hosts are each connected to the switch via a physical link. The controller is a piece of software so I decided to depict it as a box that surrounds the entire system.

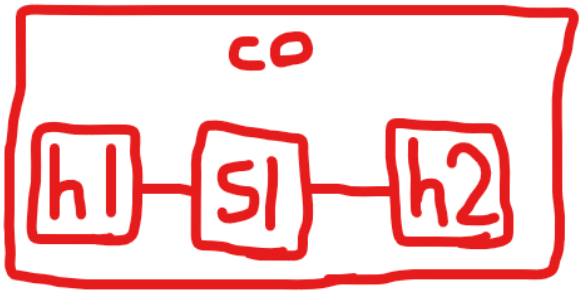

Figure 1: default topology sketch

ii. .

```
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::5cf5:deff:fe64:c21  prefixlen 64  scopeid 0x20<link>
        ether 5e:f5:de:64:0c:21  txqueuelen 1000  (Ethernet)
        RX packets 36  bytes 3862 (3.8 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 2: h1 ifconfig

```
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::945a:88ff:fe91:1009  prefixlen 64  scopeid 0x20<link>
        ether 96:5a:88:91:10:09  txqueuelen 1000  (Ethernet)
        RX packets 39  bytes 4109 (4.1 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 3: h2 ifconfig

```
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:f037:2:68dc:d25:7b38:f23b  prefixlen 64  scopeid 0x0<gl
obal>
        inet6 fe80::a826:1f3:6a15:4e79  prefixlen 64  scopeid 0x20<link>
        inet6 fd17:625c:f037:2:aac4:da9f:1252:f1bc  prefixlen 64  scopeid 0x0<g
lobal>
        ether 08:00:27:e9:5f:d1  txqueuelen 1000  (Ethernet)
        RX packets 106190  bytes 144015074 (144.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12138  bytes 805446 (805.4 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 1460  bytes 96059 (96.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1460  bytes 96059 (96.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 4: s1 ifconfig part 1

```
ovs-system: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether fa:33:15:9a:ed:ae  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 72:85:69:cb:40:42  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 12  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::80d:47ff:fe7d:1cf  prefixlen 64  scopeid 0x20<link>
        ether 0a:0d:47:7d:01:cf  txqueuelen 1000  (Ethernet)
        RX packets 14  bytes 1076 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39  bytes 4109 (4.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::6895:adff:fe7d:fe51  prefixlen 64  scopeid 0x20<link>
        ether 6a:95:ad:7d:fe:51  txqueuelen 1000  (Ethernet)
        RX packets 14  bytes 1076 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39  bytes 4109 (4.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 5: s1 ifconfig part 2

iii. As the screenshot shows, h1 only knows about 10.0.0.0, while s1 knows about the host network. This shows h1 and h2 are in isolated networks because they do not share the same routes that the switch has.

```
mininet> h1 arp -n
mininet> h1 route -n
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> s1 arp -n
Address                  HWtype  HWaddress           Flags Mask            Ifac
e
10.0.2.2                 ether   52:55:0a:00:02:02   C                     enp0
s3
mininet> s1 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.2        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
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
```

Figure 6: arp and route

b. .

```
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=5.49 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.488/5.488/5.488/0.000 ms
```

Figure 7: h1 to h2 ping

```
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=0.353 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.353/0.353/0.353/0.000 ms
```

Figure 8: h1 to h2 ping (a second time)

```
mininet> h2 ping -c 1 h1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=2.93 ms

--- 10.0.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 2.930/2.930/2.930/0.000 ms
```

Figure 9: h2 to h1 ping

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

Figure 10: pingall

c. My Ubuntu install came with Python 3.8.10 instead of 3.8.2 or 3.8.5 as shown. I'm pretty sure this is fine because the third digit just indicates a minor revision.

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

Figure 11: python version

```
ddykstra@Ubuntu:~/Desktop/Lab 0$ mn --version
2.3.1b4
```

Figure 12: mininet version

3. Part 2: Advanced Startup Options
    a. .

```
ddykstra@Ubuntu:~/Desktop/Lab 0$ sudo mn --topo single,6
*** 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 1 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=5.16 ms

--- 10.0.0.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 5.162/5.162/5.162/0.000 ms
```

Figure 13: single topology

```
ddykstra@Ubuntu:~/Desktop/Lab 0$ sudo mn --topo linear,6
*** 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,
) (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 1 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=13.1 ms

--- 10.0.0.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 13.065/13.065/13.065/0.000 ms
```

Figure 14: linear topology

The linear topology has a much higher latency because there are more links from h1 to h3. In the single topology only two links are required. In the linear network the number goes up to four, and potentially even more if we jump from h1 to h6.

b. Using the default topology with two hosts and one switch

```
ddykstra@Ubuntu:~/Desktop/Lab 0$ sudo mn --link tc,bw=10,delay=20ms
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(10.00Mbit 20ms delay) (10.00Mbit 20ms delay) (h1, s1) (10.00Mbit 20ms delay) (
10.00Mbit 20ms delay) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...(10.00Mbit 20ms delay) (10.00Mbit 20ms delay)
*** Starting CLI:
mininet> h1 ping -c5 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=188 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=102 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=89.7 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=97.5 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=90.7 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4015ms
rtt min/avg/max/mdev = 89.701/113.443/187.504/37.296 ms
```

Figure 15: network with a 20ms delay

Theoretically there should be at least 80ms of delay, and we see that in the screenshot.

c. .

```
ddykstra@Ubuntu:~/mininet/custom$ sudo mn --custom ~/mininet/custom/topo-2sw-2h
ost.py --topo mytopo --test pingall
*** 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
```

Figure 16: custom topology

```
c0
*** Stopping 3 links
...
*** Stopping 2 switches
s3 s4
*** Stopping 2 hosts
h1 h2
*** Done
completed in 5.748 seconds
```

Figure 17: ending the custom topology

This network has two hosts, and two switches, there are three links tying them all together, each host gets a switch, and the two switches are tied together.

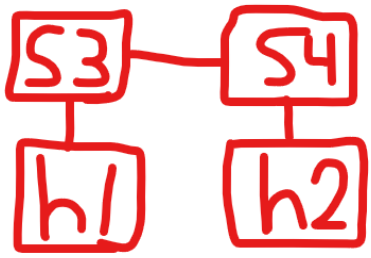

Figure 18: sketch of custom network topology