Lab 6

Computer Networks Lab (CS302)

Team Members:

Alan Tony (191CS207)

Aashish Prateek (191CS225)

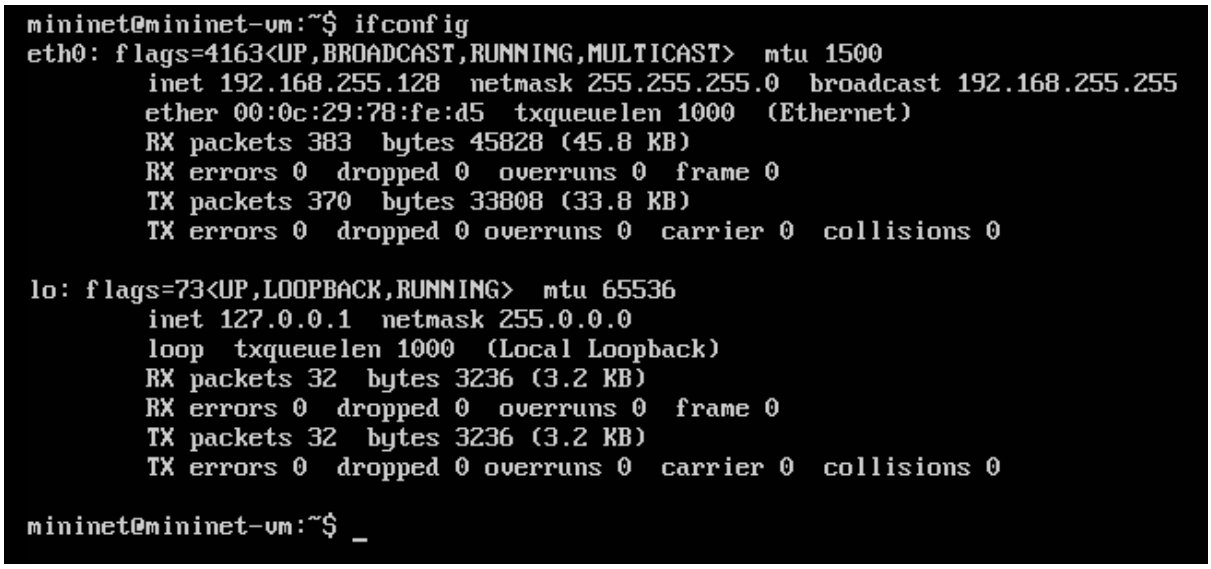
Steps followed:

1. Mininet was imported from *.ovf* file and installed in VMware Workstation Player.
2. Ubuntu VM was booted from VMware Workstation Player
3. Logged into Mininet

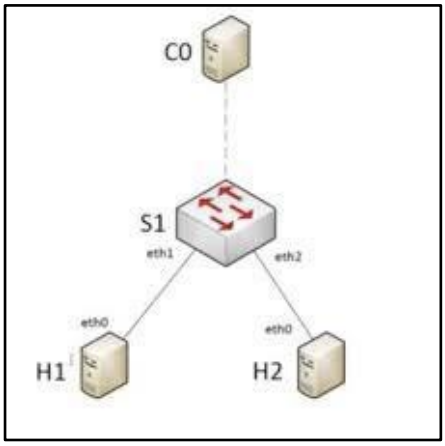
Pre-question Trials

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.

To experiment with the network, we need to *ssh* into the Mininet VM. We get the IP address of the VM by entering *ifconfig* on the Mininet terminal. Also pass the -X option to open an *xterm* interface to use terminals of different hosts in the network on our Ubuntu VM.



To start interacting with the network we initiate the network using command '*sudo mn'.* The network simulated by this command is of Mininet's default minimal topology which looks like:



Mininet's minimal topology

- **ping**:

```
mininet@mininet-vm:~$ ping www.google.com
PING www.google.com (142.250.195.36) 56(84) bytes of data.
64 bytes from maa03s37-in-f4.1e100.net (142.250.195.36): icmp_seq=1 ttl=128 time=77.0 ms
64 bytes from maa03s37-in-f4.1e100.net (142.250.195.36): icmp_seq=2 ttl=128 time=15.1 ms
64 bytes from maa03s37-in-f4.1e100.net (142.250.195.36): icmp_seq=3 ttl=128 time=16.1 ms
64 bytes from maa03s37-in-f4.1e100.net (142.250.195.36): icmp_seq=4 ttl=128 time=15.0 ms
64 bytes from maa03s37-in-f4.1e100.net (142.250.195.36): icmp_seq=5 ttl=128 time=16.0 ms
^C
--- www.google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4012ms
rtt min/avg/max/mdev = 15.068/27.902/77.037/24.571 ms
mininet@mininet-vm:~$ ping -c3 www.google.com
PING www.google.com (142.250.195.228) 56(84) bytes of data.
64 bytes from maa03s43-in-f4.1e100.net (142.250.195.228): icmp_seq=1 ttl=128 time=73.3 ms
64 bytes from maa03s43-in-f4.1e100.net (142.250.195.228): icmp_seq=2 ttl=128 time=55.0 ms
64 bytes from maa03s43-in-f4.1e100.net (142.250.195.228): icmp_seq=3 ttl=128 time=25.6 ms

--- www.google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 25.633/51.339/73.327/19.647 ms
mininet@mininet-vm:~$
```

- **arp**: Using the arp command allows **you to display and modify the Address Resolution Protocol (ARP) cache**

```
mininet@mininet-vm:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.255.129          ether   00:0c:29:e4:b6:a1   C                     eth0
_gateway                 ether   00:50:56:f6:8c:8c   C                     eth0
192.168.255.254          ether   00:50:56:fd:aa:3c   C                     eth0
192.168.255.1            ether   00:50:56:c0:00:08   C                     eth0
mininet@mininet-vm:~$
```

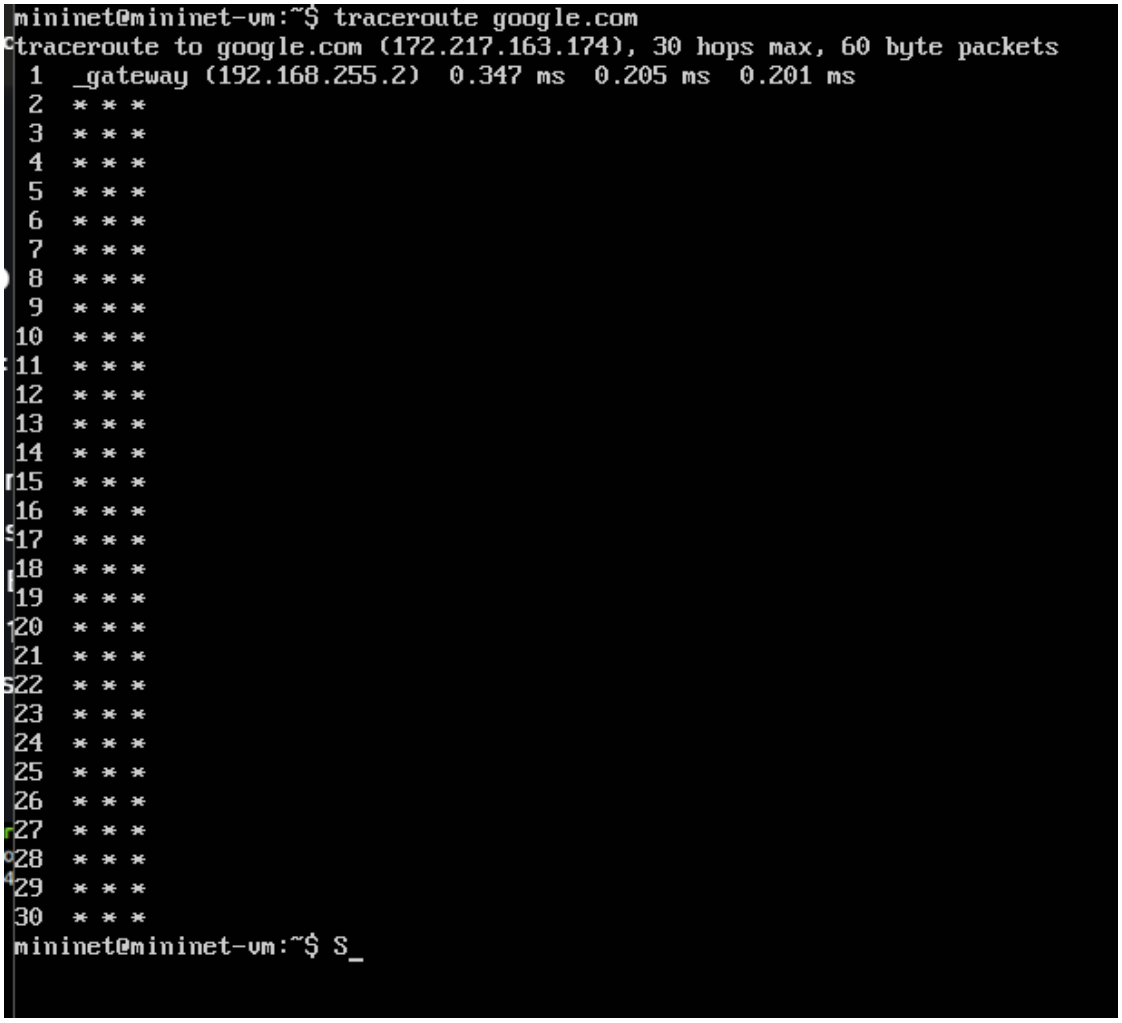
- **nslookup**: Nslookup (stands for "Name Server Lookup") is a **useful command for getting information from DNS server**.

```
mininet@mininet-vm:~$ nslookup
> set all
Default server: 127.0.0.53
Address: 127.0.0.53#53

Set options:
  novc                  nodebug         nod2
  search                recurse
  timeout = 0           retry = 3       port = 53       ndots = 1
  querytype = A         class = IN
  srchlist = localdomain
> www.google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   www.google.com
Address: 142.250.205.228
Name:   www.google.com
Address: 2404:6800:4007:828::2004
>
```

- **whois**: Whois is a **widely used Internet record listing that identifies who owns a domain and how to get in contact with them**.

```
mininet@mininet-vm:~$ whois google.com
   Domain Name: GOOGLE.COM
   Registry Domain ID: 2138514_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.markmonitor.com
   Registrar URL: http://www.markmonitor.com
   Updated Date: 2019-09-09T15:39:04Z
   Creation Date: 1997-09-15T04:00:00Z
   Registry Expiry Date: 2028-09-14T04:00:00Z
   Registrar: MarkMonitor Inc.
   Registrar IANA ID: 292
   Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
   Registrar Abuse Contact Phone: +1.2083895740
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
```

- **traceroute**: Traceroute is **a command that runs tools used for network diagnostics**. These tools trace the paths data packets take from their source to their destinations, allowing administrators to better resolve connectivity issues.



- **hostname**: Get or set hostname or DNS domain name

- **route**: In computing, route is a command used to view and manipulate the IP routing table in Unix-like and Microsoft Windows operating systems and also in IBM OS/2 and ReactOS.

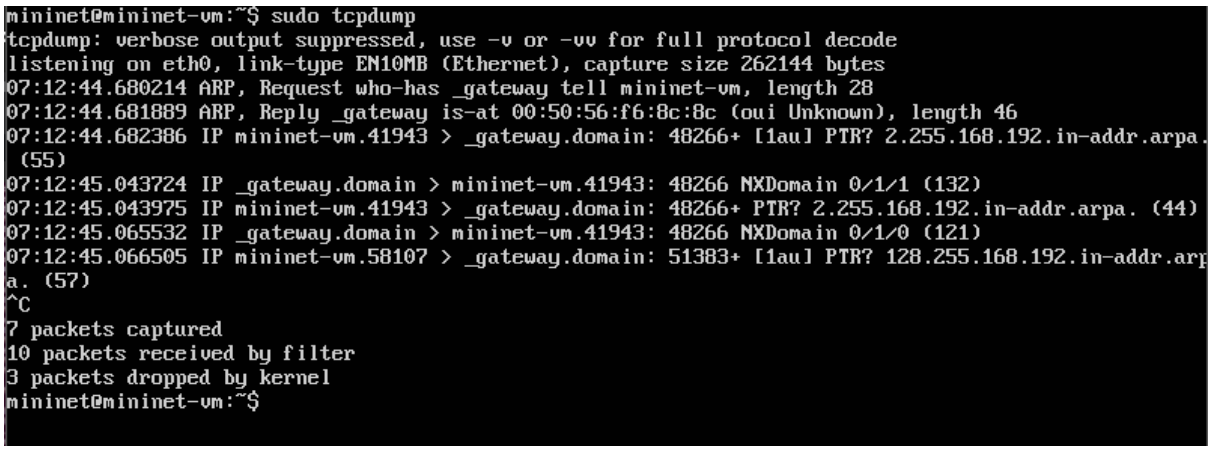
```
mininet@mininet-vm:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         _gateway        0.0.0.0         UG    100    0        0 eth0
192.168.255.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
_gateway        0.0.0.0         255.255.255.255 UH    100    0        0 eth0
mininet@mininet-vm:~$
```

- **netstat**: In computing, netstat is a command-line network utility that displays network connections for Transmission Control Protocol, routing tables, and a number of network interface and network protocol statistics.

```
mininet@mininet-vm:~$ netstat -e -s
Ip:
    Forwarding: 2
    1771 total packets received
    2 with invalid addresses
    0 forwarded
    0 incoming packets discarded
    1769 incoming packets delivered
    1862 requests sent out
Icmp:
    3 ICMP messages received
    0 input ICMP message failed
    ICMP input histogram:
        timeout in transit: 3
    0 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
IcmpMsg:
        InType11: 3
Tcp:
    719 active connection openings
    0 passive connection openings
    716 failed connection attempts
    0 connection resets received
    0 connections established
    1458 segments received
    1461 segments sent out
    0 segments retransmitted
    0 bad segments received
    716 resets sent
Udp:
    307 packets received
    0 packets to unknown port received
    0 packet receive errors
    401 packets sent
    0 receive buffer errors
    0 send buffer errors
```

- **tcpdump**: tcpdump is a packet analyzer that is launched from the command line. It can be used **to analyze network traffic by**

intercepting and displaying packets that are being created or received by the computer it's running on.



P.T.O

Virtual Network Setup

Here we are beginning the network using the ‘sudo mn’ command from the host virtual machine which connects to the mininet VM.

```
alan@alan-virtual-machine:~$ ssh -X mininet@192.168.255.128
mininet@192.168.255.128's password:
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-112-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Oct 27 06:59:30 2021
mininet@mininet-vm:~$ sudo 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> S
```

The ‘nodes’ command lets us view the details of all the nodes or components of the network.

```
mininet> nodes
available nodes are:
c0 h1 h2 s1
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet> h1 ping -c3 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=47.4 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.409 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.075 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2030ms
rtt min/avg/max/mdev = 0.075/15.994/47.498/22.277 ms
mininet>
```

Answers

PART A

1.

i. IP addresses: The ‘dump’ command is used within the mininet framework to get a list of all the network elements along with their IP addresses.

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2251>
<Host h2: h2-eth0:10.0.0.2 pid=2253>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2258>
<Controller c0: 127.0.0.1:6653 pid=2244>
```

ii. MAC addresses and interfaces (obtained with *ifconfig*): ‘ifconfig’ is used to obtain all configuration-related information regarding the virtual network.

iii.

- h1
    a. Interfaces for ‘h1’ include ‘h1-eth0’ whose MAC address is 00:00:00:00:00:01
    b. another interface for h1 is ‘lo’ whose IP address is 127.0.0.1 and is also known as a loopback interface.

```
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
        ether 00:00:00:00:00:01  txqueuelen 1000  (Ethernet)
        RX packets 20  bytes 1624 (1.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1624 (1.6 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
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 16  bytes 1472 (1.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16  bytes 1472 (1.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

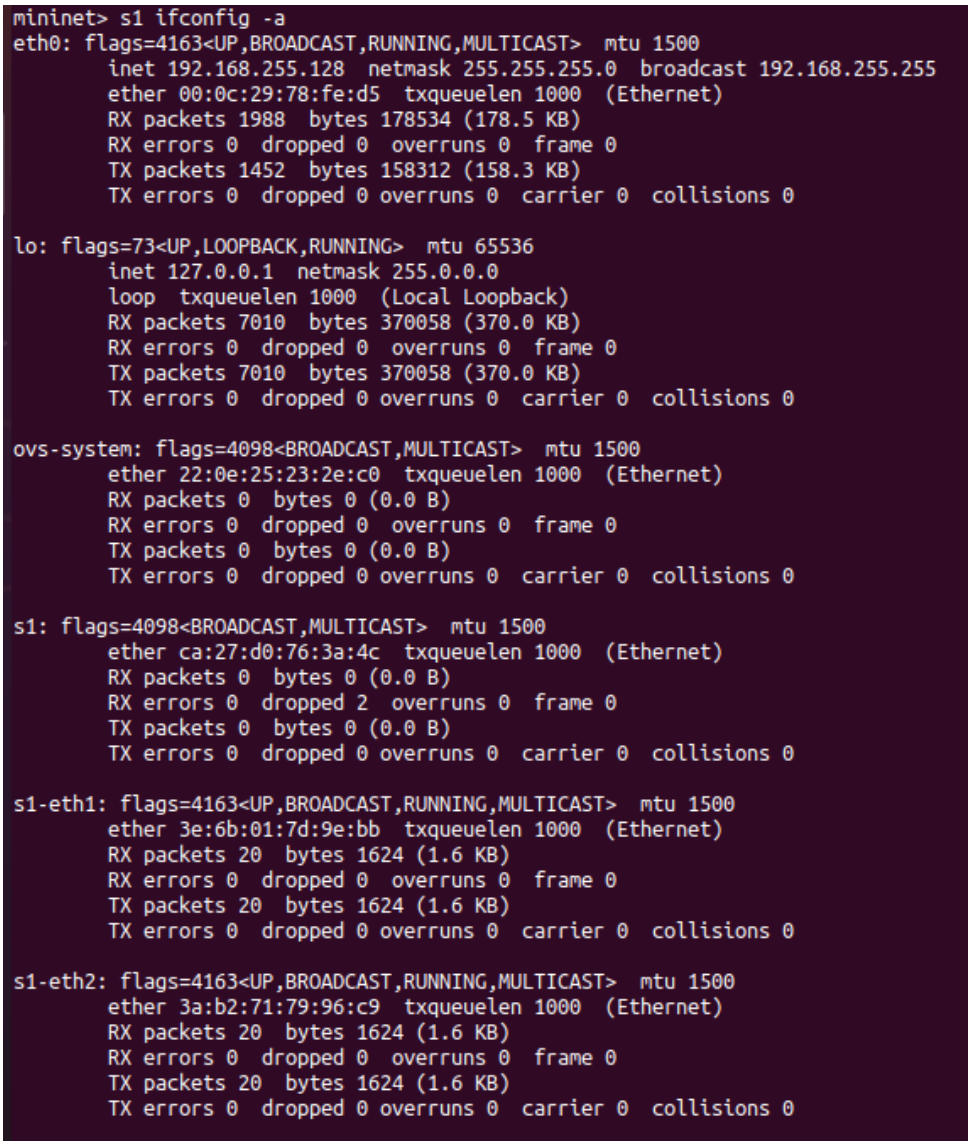
- h2
    a. Interfaces for ‘h2’ includes ‘h2-eth0’ whose MAC address is 00:00:00:00:00:02
    b. another interface for h2 is ‘lo’ whose IP address is 127.0.0.1 and is also known as a loopback interface.

```
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
        ether 00:00:00:00:00:02  txqueuelen 1000  (Ethernet)
        RX packets 20  bytes 1624 (1.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1624 (1.6 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
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8  bytes 736 (736.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 736 (736.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- c0: List of interfaces for ‘c0’ or basically the controller includes:
    c. ‘eth-0’ with MAC address 00:0c:29:78:fe:d5
    d. loopback(lo) with IP 127.0.0.1
    e. ‘ovs-system’ with MAC address 22:0e:25:23:2e:c0
    f. ‘s1’ with MAC address ca:27:d0:76:3a:4c
    g. ‘s1-eth1’ with MAC address 3e:6b:01:7d:9e:bb
    h. ‘s1-eth2’ with MAC address 3a:b2:71:79:96:c9

```
mininet> c0 ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.255.128  netmask 255.255.255.0  broadcast 192.168.
        ether 00:0c:29:78:fe:d5  txqueuelen 1000  (Ethernet)
        RX packets 1872  bytes 169114 (169.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1372  bytes 147268 (147.2 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
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 6953  bytes 366790 (366.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6953  bytes 366790 (366.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ovs-system: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 22:0e:25:23:2e:c0  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 ca:27:d0:76:3a:4c  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 2  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
        ether 3e:6b:01:7d:9e:bb  txqueuelen 1000  (Ethernet)
        RX packets 20  bytes 1624 (1.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1624 (1.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 3a:b2:71:79:96:c9  txqueuelen 1000  (Ethernet)
        RX packets 20  bytes 1624 (1.6 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 1624 (1.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- s1: The list of interfaces for ‘s1’ include:
    i. ‘eth-0’ with MAC address 00:0c:29:78:fe:d5
    j. loopback(lo) with IP 127.0.0.1
    k. ‘ovs-system’ with MAC address 22:0e:25:23:2e:c0
    l. ‘s1’ with MAC address ca:27:d0:76:3a:4c
    m. ‘s1-eth1’ with MAC address 3e:6b:01:7d:9e:bb
    n. ‘s1-eth2’ with MAC address 3a:b2:71:79:96:c9



2. The latency between mininet vm and :

   a. www.rutgers.edu : Network latency, sometimes called lag, is the term used to describe delays in communication over a network. Latency meaning in networking is best thought of as the amount of time it takes for a packet of data to be captured, transmitted, processed through multiple devices, then received at its destination and decoded. Here we observe that the network latency between 'www.rutgers.edu' and mininet VM is around the 250ms range.

```
mininet@mininet-vm:~$ ping -c10 www.rutgers.edu
PING www.rutgers.edu (128.6.46.88) 56(84) bytes of data.
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=1 ttl=128 time=292 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=2 ttl=128 time=249 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=3 ttl=128 time=243 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=4 ttl=128 time=243 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=5 ttl=128 time=243 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=6 ttl=128 time=243 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=7 ttl=128 time=243 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=8 ttl=128 time=249 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=9 ttl=128 time=245 ms
64 bytes from www-new.rutgers.edu (128.6.46.88): icmp_seq=10 ttl=128 time=244 ms

--- www.rutgers.edu ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9026ms
rtt min/avg/max/mdev = 243.127/249.758/292.268/14.365 ms
```

   b. www.stanford.edu :

   Here we observe that the network latency between 'www.stanford.com' and mininet VM is around the 13ms range.

```
mininet@mininet-vm:~$ ping -c10 www.stanford.edu
PING pantheon-systems.map.fastly.net (151.101.158.133) 56(84) bytes of data.
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=1 ttl=128 time=62.1 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=2 ttl=128 time=20.0 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=3 ttl=128 time=13.2 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=4 ttl=128 time=13.7 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=5 ttl=128 time=14.0 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=6 ttl=128 time=13.6 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=7 ttl=128 time=13.9 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=8 ttl=128 time=13.5 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=9 ttl=128 time=13.7 ms
64 bytes from 151.101.158.133 (151.101.158.133): icmp_seq=10 ttl=128 time=18.5 ms

--- pantheon-systems.map.fastly.net ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9018ms
rtt min/avg/max/mdev = 13.238/19.668/62.189/14.348 ms
```

c. www.google.com :

Here we observe that the network latency between 'www.google.com' and mininet VM is around the 13ms range.

```
mininet@mininet-vm:~$ ping -c10 www.google.com
PING www.google.com (142.250.205.228) 56(84) bytes of data.
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=1 ttl=128 time=65.5 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=2 ttl=128 time=12.9 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=3 ttl=128 time=14.2 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=4 ttl=128 time=12.4 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=5 ttl=128 time=13.4 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=6 ttl=128 time=12.9 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=7 ttl=128 time=13.2 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=8 ttl=128 time=12.8 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=9 ttl=128 time=13.0 ms
64 bytes from maa05s28-in-f4.1e100.net (142.250.205.228): icmp_seq=10 ttl=128 time=12.7 ms

--- www.google.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9028ms
rtt min/avg/max/mdev = 12.420/18.349/65.532/15.735 ms
```

finally we make conclusions that the average latency between mininet and the respective sites is in the order

www.google.com < www.stanford.com < www.rudgers.edu

here www.rudgers.edu has the highest average latency and network delay. hence its lag is the highest and www.google.com has the least delay or lag. www.stanford.com falls very close to www.google.com in terms of network latency or lag.

PART B

a. Print the MAC address of host h1. Print the MAC addresses of switch s1. Explain the different interfaces that s1 has

i. MAC address of host h1:

```
mininet> h1 ifconfig
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
        ether 00:00:00:00:00:01  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

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        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
```

ii. MAC address of host s1 (contained in s1 section, in flags) and different interfaces that s1 has (the interfaces are eth0, eth1, and eth2):

```
mininet> s1 ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.255.128  netmask 255.255.255.0  broadcast 192.168.255.255
        ether 00:0c:29:78:fe:d5  txqueuelen 1000  (Ethernet)
        RX packets 895  bytes 88965 (88.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 738  bytes 75284 (75.2 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
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 5541  bytes 280516 (280.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5541  bytes 280516 (280.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ovs-system: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 22:0e:25:23:2e:c0  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 ca:27:d0:76:3a:4c  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-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 3e:6b:01:7d:9e:bb  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-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 3a:b2:71:79:96:c9  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
```

b. The ‘ping’ command lets us ping h1 from h2. to view the ARP entries, we can observe the ping statistics column where we can see that h1’s IP address

to MAC address is stored in the ARP entry table of h2. to view this table we use the ‘h1 arp -a’ command. and this same logic works vice-versa too.

```
mininet> h1 ping -c3 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=1.57 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.074 ms

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2033ms
rtt min/avg/max/mdev = 0.073/0.572/1.570/0.705 ms
mininet> h1 arp -a
? (10.0.0.2) at 00:00:00:00:00:02 [ether] on h1-eth0
mininet> h2 ping -c3 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=1.75 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.075 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.081 ms

--- 10.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2021ms
rtt min/avg/max/mdev = 0.075/0.637/1.756/0.791 ms
mininet> h2 arp -a
? (10.0.0.1) at 00:00:00:00:00:01 [ether] on h2-eth0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=2251>
<Host h2: h2-eth0:10.0.0.2 pid=2253>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=2258>
<Controller c0: 127.0.0.1:6653 pid=2244>
mininet>
```

3. The following screenshot shows the TCP throughput from h1 to h2. We observe that the bandwidth between h1 and h2 is 19.8 Gbits/sec

```
mininet@mininet-vm:~$ sudo 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: ['19.8 Gbits/sec', '19.8 Gbits/sec']
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 8.503 seconds
```

******